



## Single-cell ATAC-seq workflow

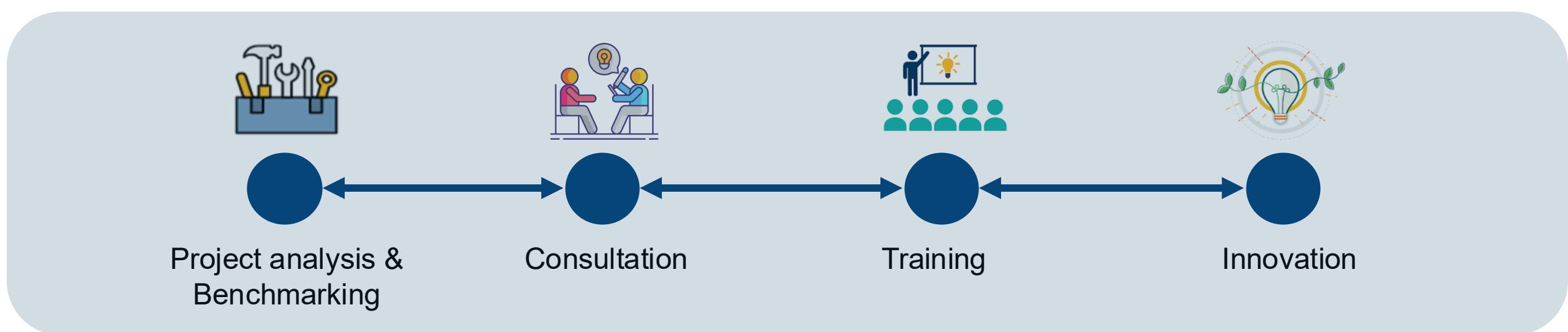
Antonia (Tonia) Chroni, PhD  
Senior Bioinformatics Research Scientist



# Bioinformatics core, Department of Developmental Neurobiology



**Providing advanced bioinformatic services for investigators to leverage omics data**



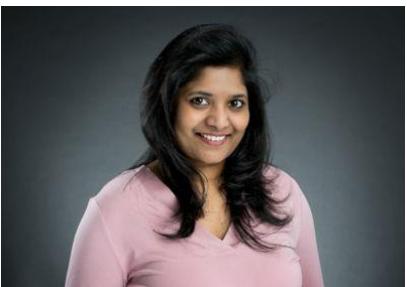
**Cody Alexander Ramirez, PhD**  
Senior Bioinformatics Research Scientist  
Core Director  
Boston, Massachusetts



**Antonia Chroni, PhD**  
Senior Bioinformatics Research Scientist  
New York, New York



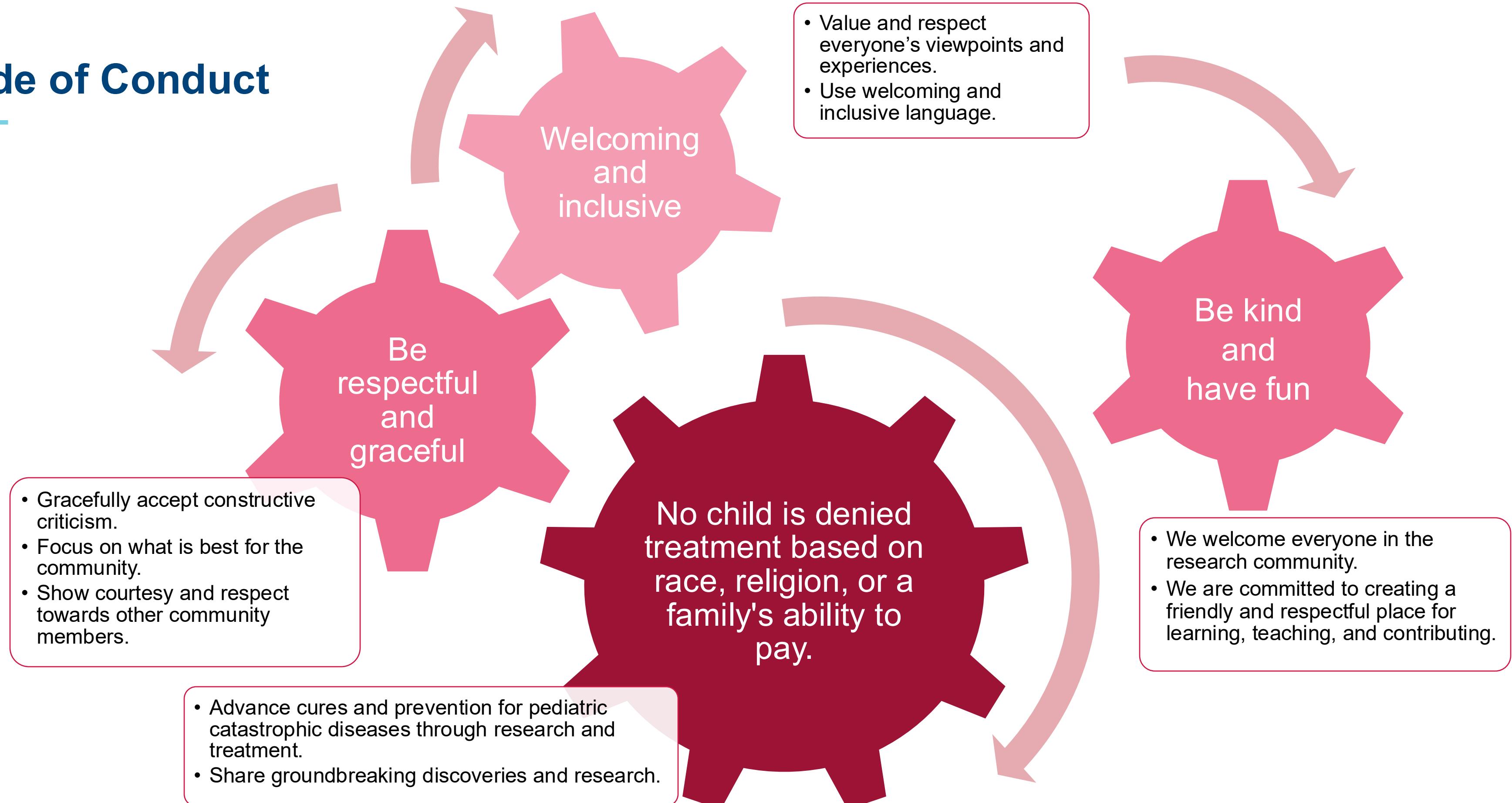
**Sharon Freshour, PhD**  
Bioinformatics Research Scientist  
Tacoma, Washington



**Asha Jacob Jannu**  
Bioinformatics Research Scientist  
Indianapolis, Indiana



# Code of Conduct



# Learning Objectives

1. Overview of Tools and Methods in the `scEpiGenie` workflow
2. Reviewing Results, Plots, and Reports



# Single cell ATAC Seq Epigenie workflow (scEpiGenie)

---

- 🚀 Automated suite of analysis modules for upstream and downstream analyses
- 🧬 scATAC data analysis from 10X sequencing technology
- 🧬 Human and mouse genome experiments
- 📝 Thorough and step-by-step documentation on how to use each analysis module
- 👉 Reproducible and replicable results via Docker/singularity
- 🌐 Publicly available at [stjude-dnb-binfcore/sc-epigenie](https://stjude-dnb-binfcore/sc-epigenie)



# scEpiGenie: A Companion to Snap

---

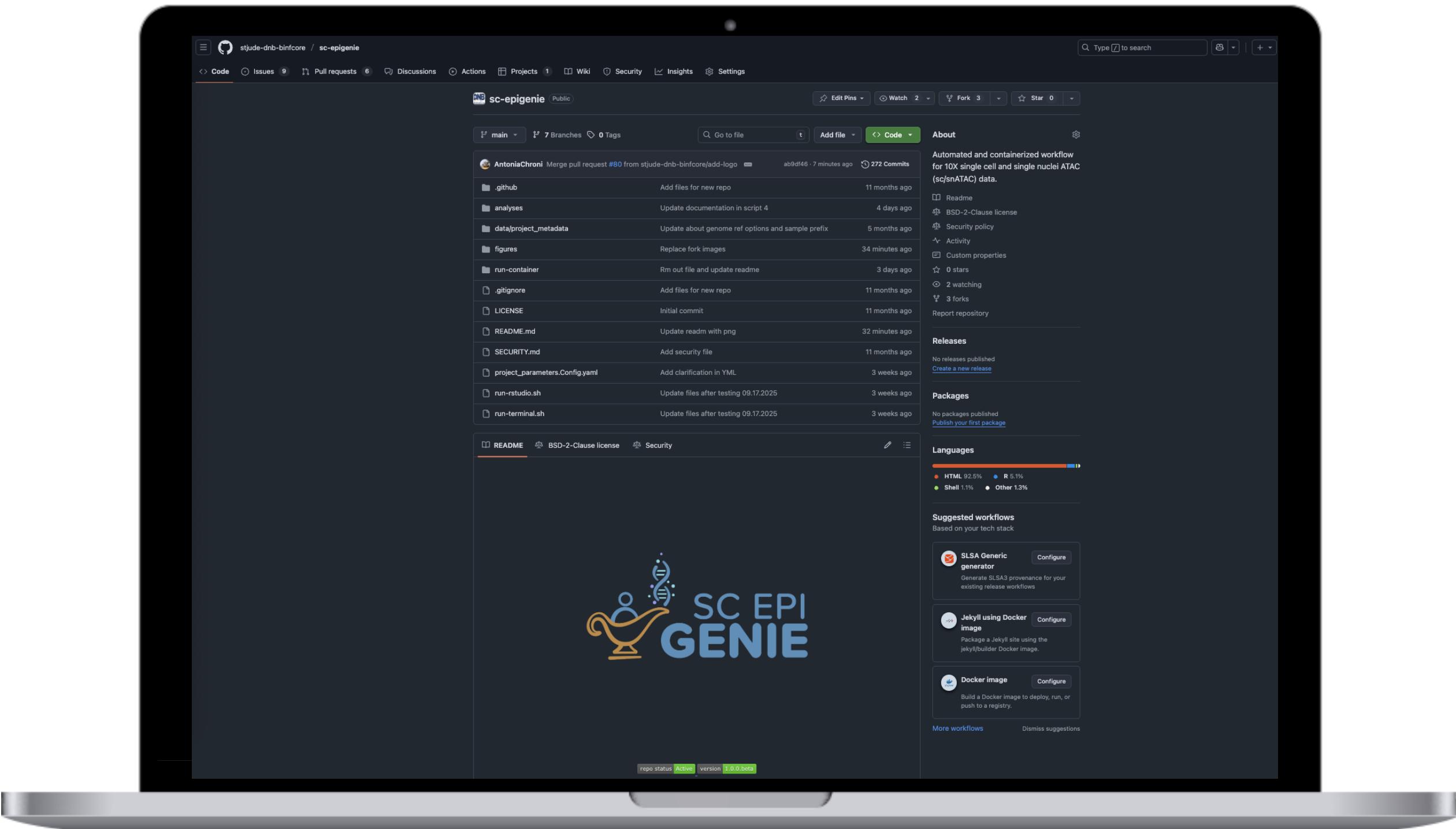
- scEpiGenie is a companion pipeline to Snap, designed for matched scRNA-seq data analysis.
- Snap pipeline is publicly available at GitHub:



[stjude-dnb-binfcore/sc-rna-seq-snap](https://github.com/stjude-dnb-binfcore/sc-rna-seq-snap)



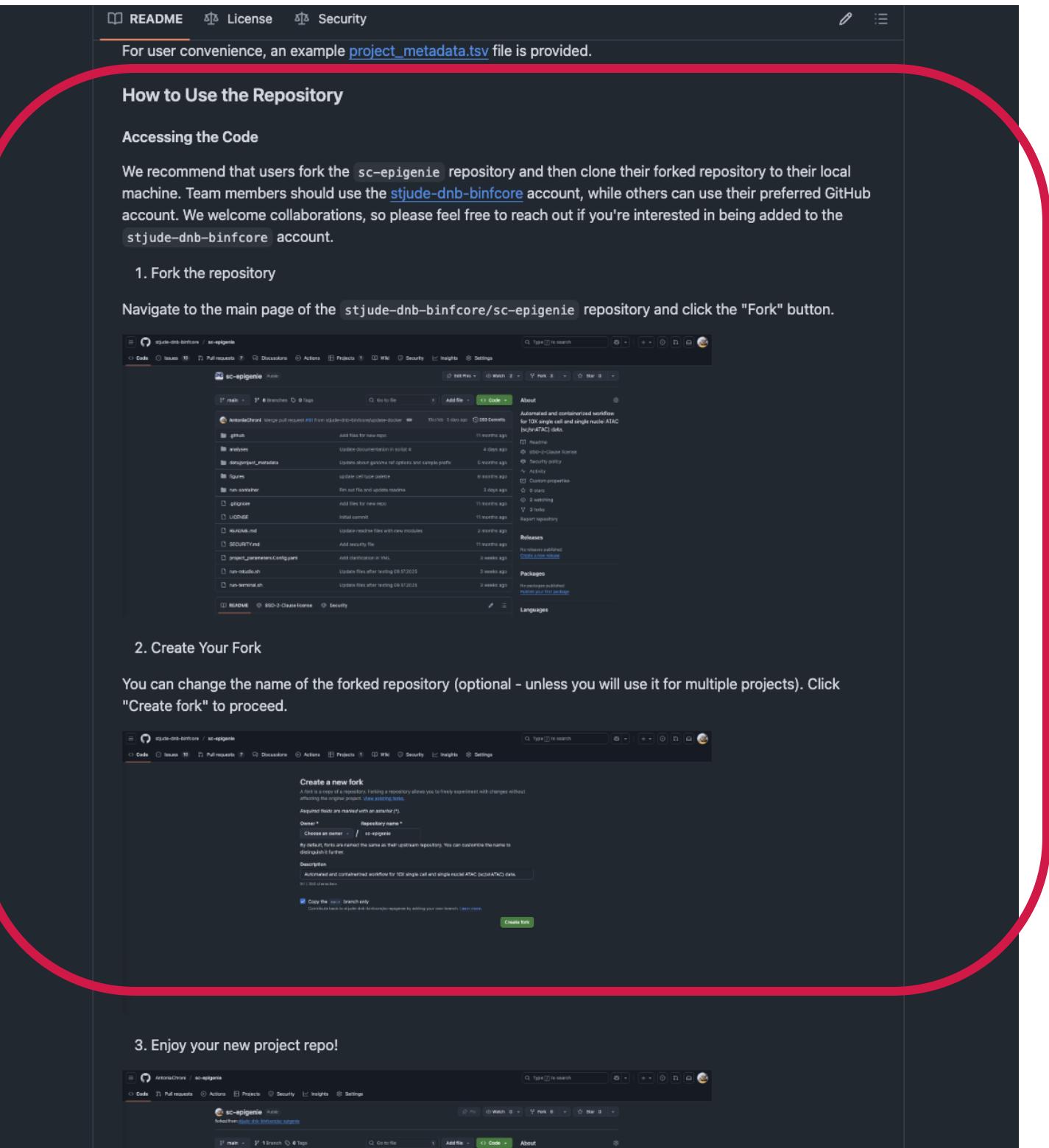
# Single cell ATAC Seq Epigenie workflow (scEpiGenie)



<https://github.com/stjude-dnb-binfcore/sc-epigenie>



# How to access the code



For user convenience, an example [project\\_metadata.tsv](#) file is provided.

## How to Use the Repository

### Accessing the Code

We recommend that users fork the `sc-epigenie` repository and then clone their forked repository to their local machine. Team members should use the `stjude-dnb-bincore` account, while others can use their preferred GitHub account. We welcome collaborations, so please feel free to reach out if you're interested in being added to the `stjude-dnb-bincore` account.

- 1. Fork the repository**

Navigate to the main page of the `stjude-dnb-bincore/sc-epigenie` repository and click the "Fork" button.

**2. Create Your Fork**

You can change the name of the forked repository (optional - unless you will use it for multiple projects). Click "Create fork" to proceed.

- 3. Enjoy your new project repo!**

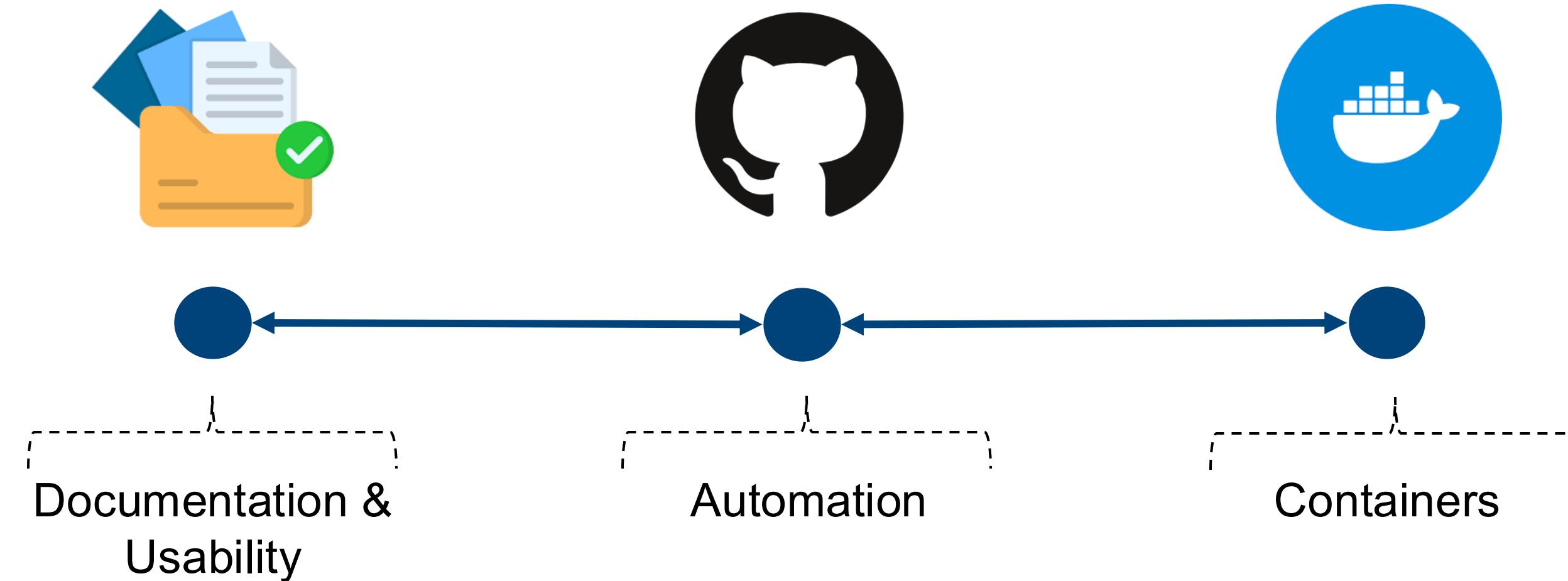


# scEpiGenie is automated, reproducible and replicable

---



# scEpiGenie is automated, reproducible and replicable

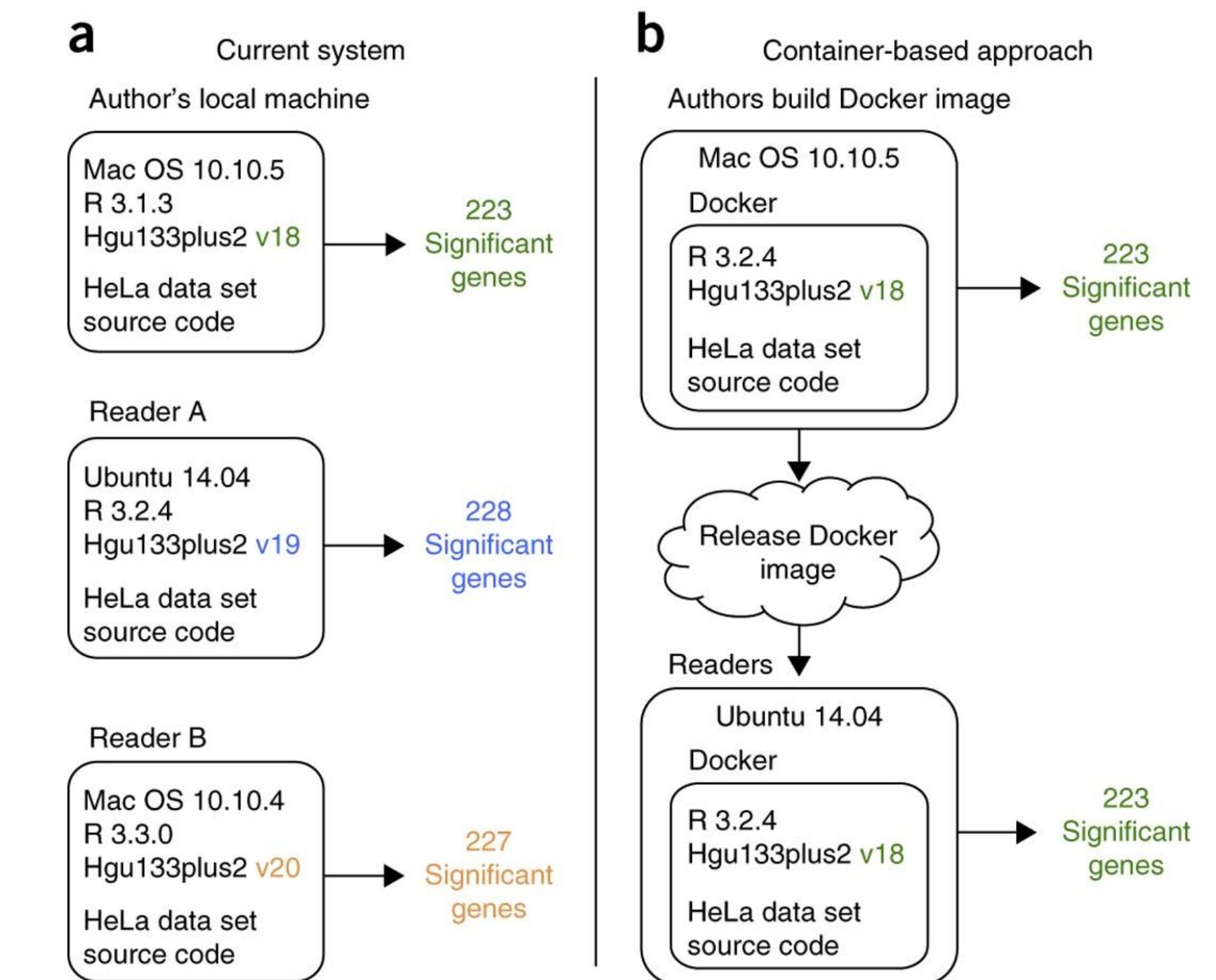


Slide from the "[Automation and Reproducibility in Computational Biology](#)" course



# scEpiGenie is automated, reproducible and replicable

- Docker and Singularity
  - “Containers” that include everything from the operating system up
  - Run one OS inside another, with all the things frozen to particular versions.
- Docker image
  - `achronistjude/singlecell-r4.4-seurat4.4-signac1.16`



Slide from the "[Automation and Reproducibility in Computational Biology](#)" course





# DevOps Containers

---

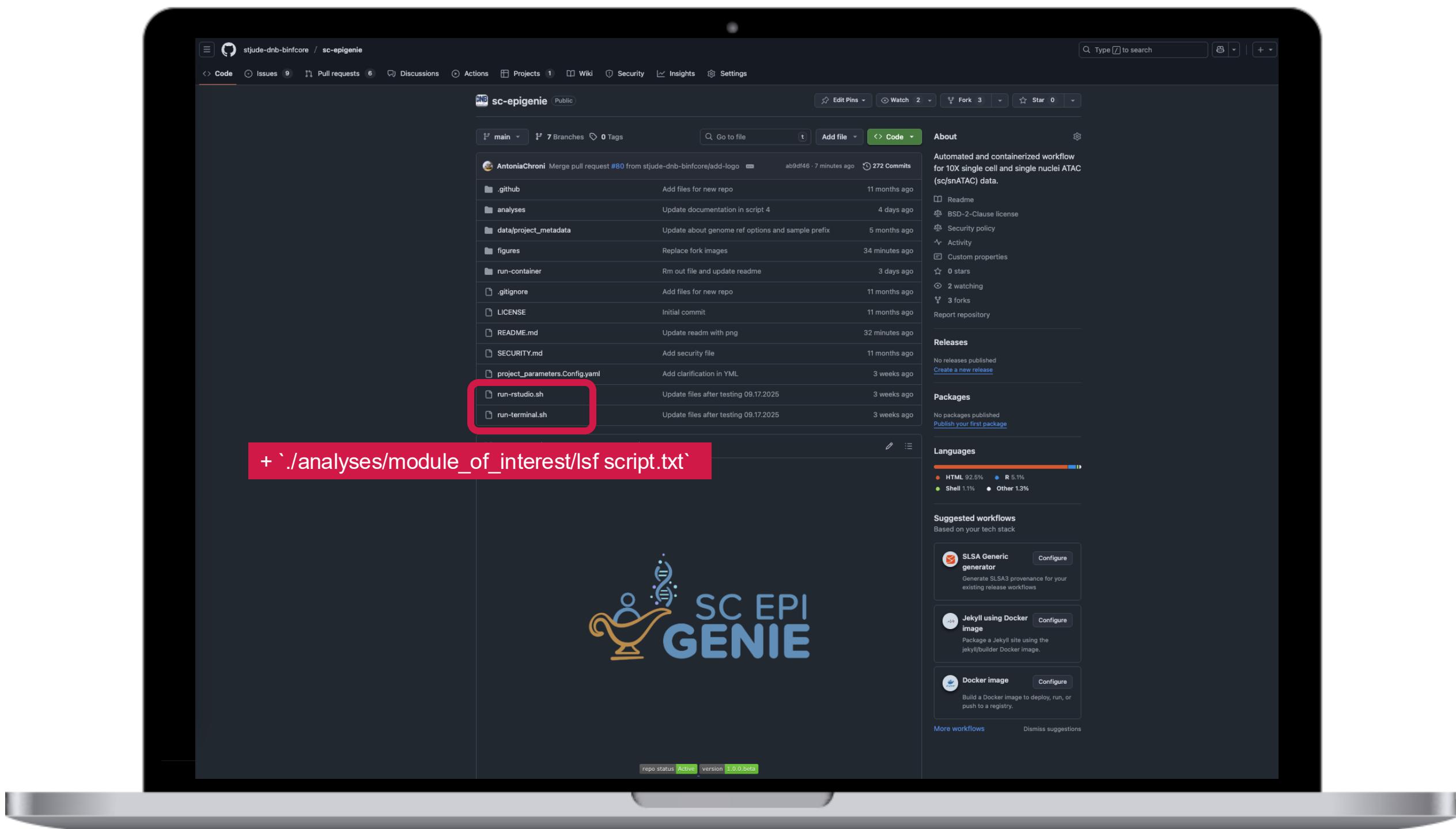
- A collection of reusable container images designed to support various stages of data processing pipelines, i.e., scRNA-seq and scATAC-seq
- Each container is self-contained and portable, making it easy to use them independently in other projects or custom workflows.
- DevOps Containers is publicly available at GitHub:



[stjude-dnb-binfcore/devops-containers](https://github.com/stjude-dnb-binfcore/devops-containers)



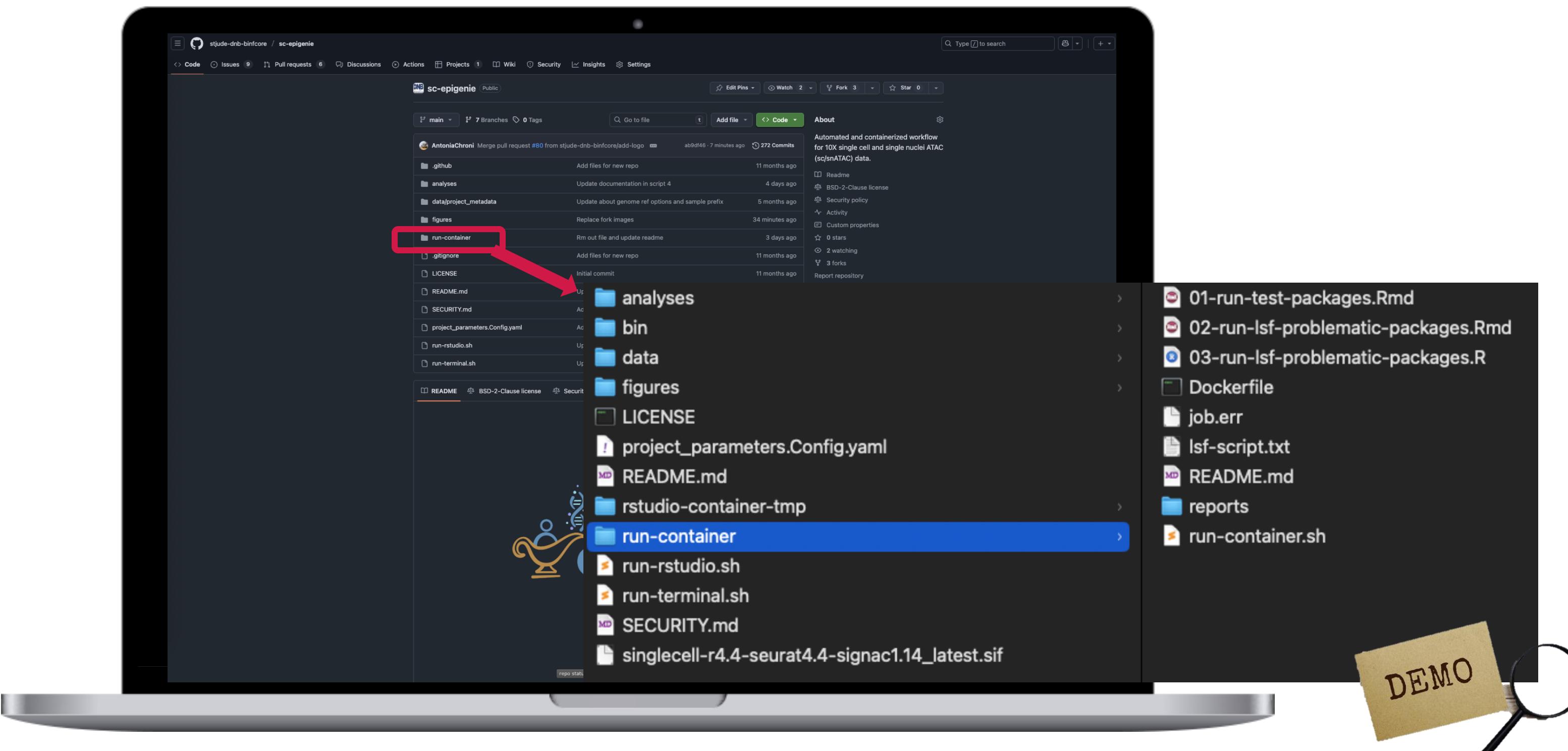
# scEpiGenie is automated, reproducible and replicable



<https://github.com/stjude-dnb-binfcore/sc-epigenie>



# scEpiGenie is automated, reproducible and replicable





# Repository organization and project tidiness

---

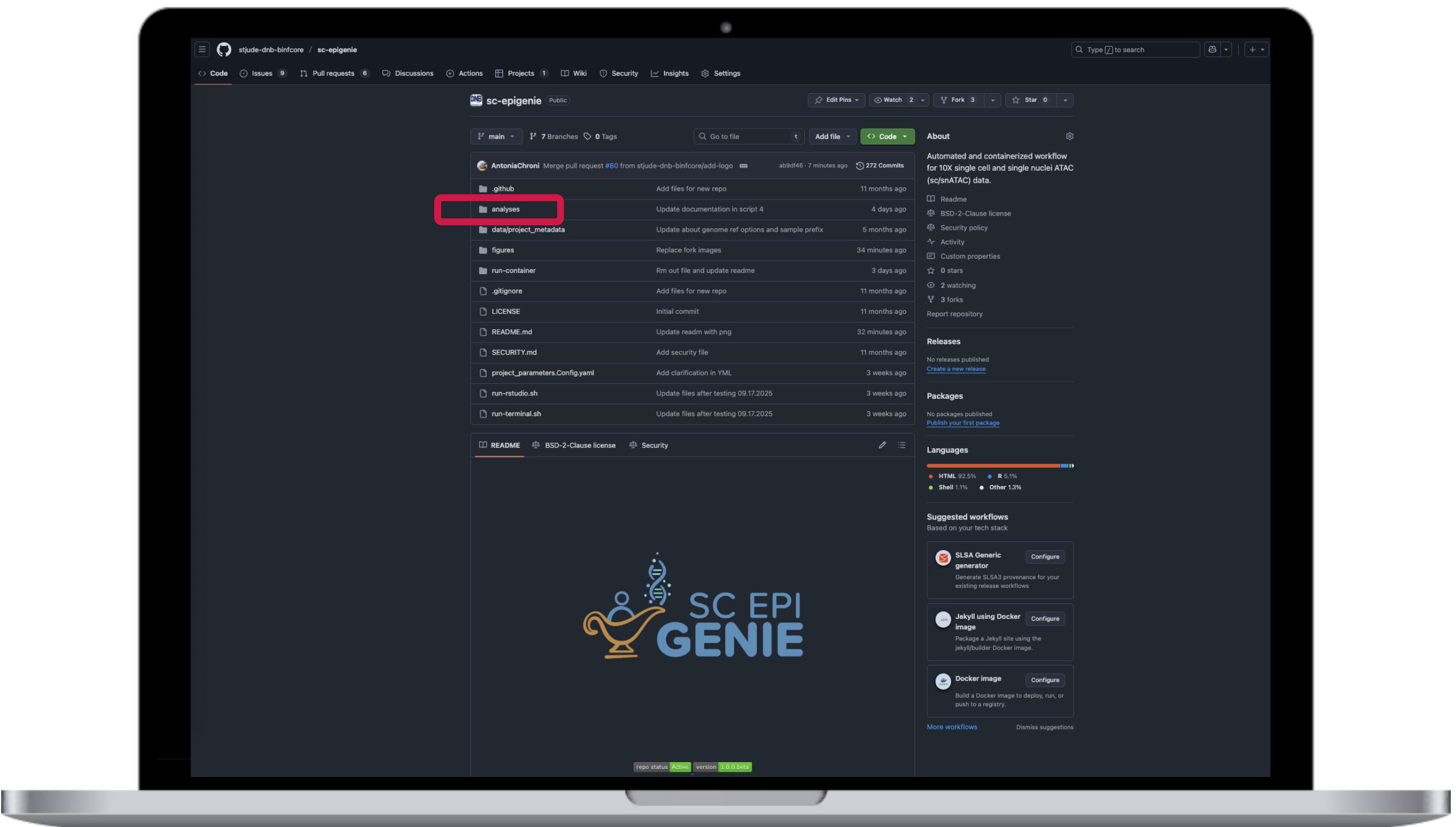
- **Use Folders/Directories**
- **Keep separate projects separated**
- **Separate sections for units within a project**
  - analyses/<module\_name>
    - code
    - plots
    - results
  - data
  - figures
  - LICENSE
  - project\_parameters.Config.yaml
  - SECURITY.md
  - ...
- **Documentation throughout:** Describe what files do and how they are organized/folder.

Slide from the "[Automation and Reproducibility in Computational Biology](#)" course





# Repository organization and project tidiness

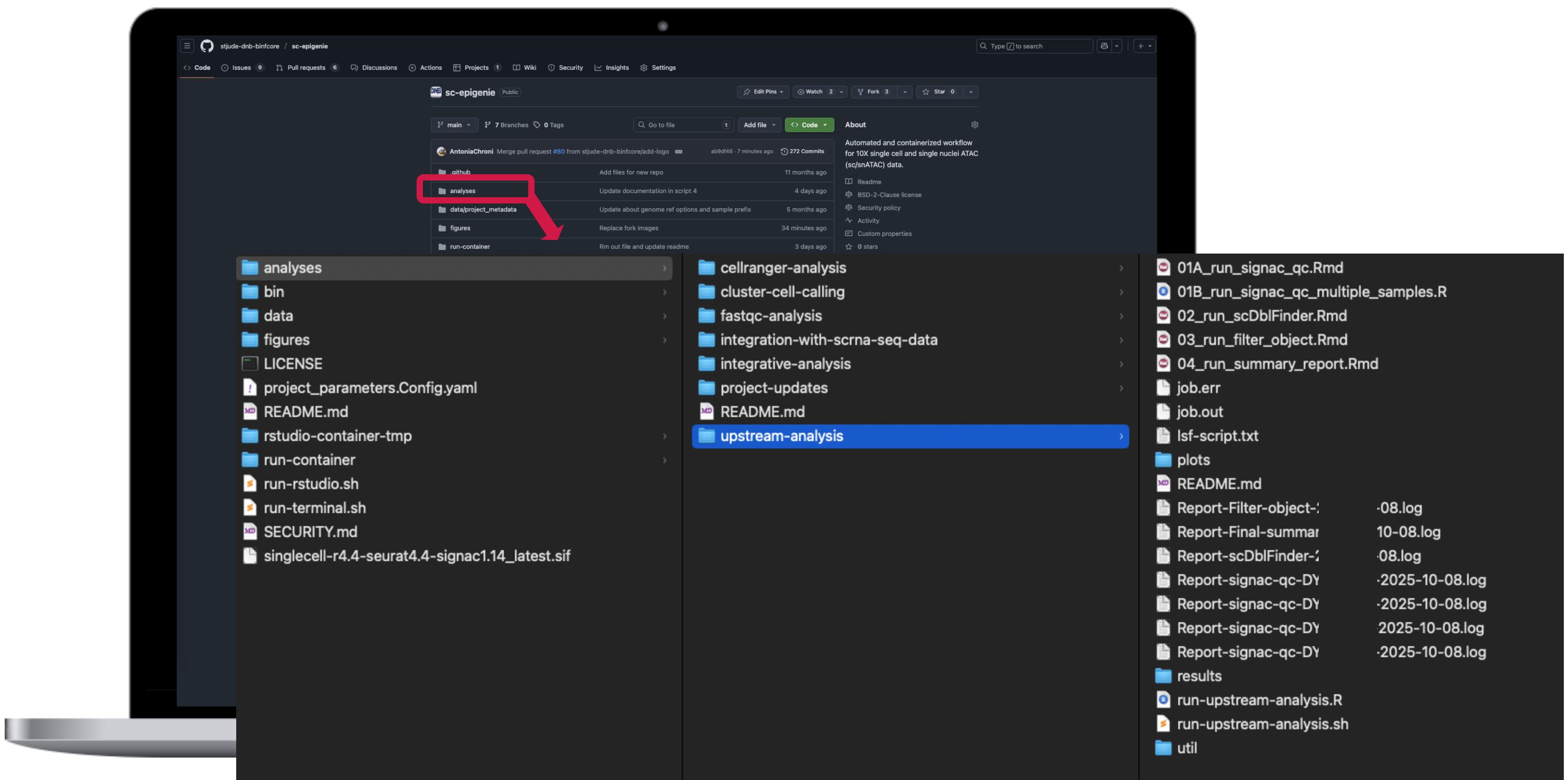


<https://github.com/stjude-dnb-binfcore/sc-epigenie>

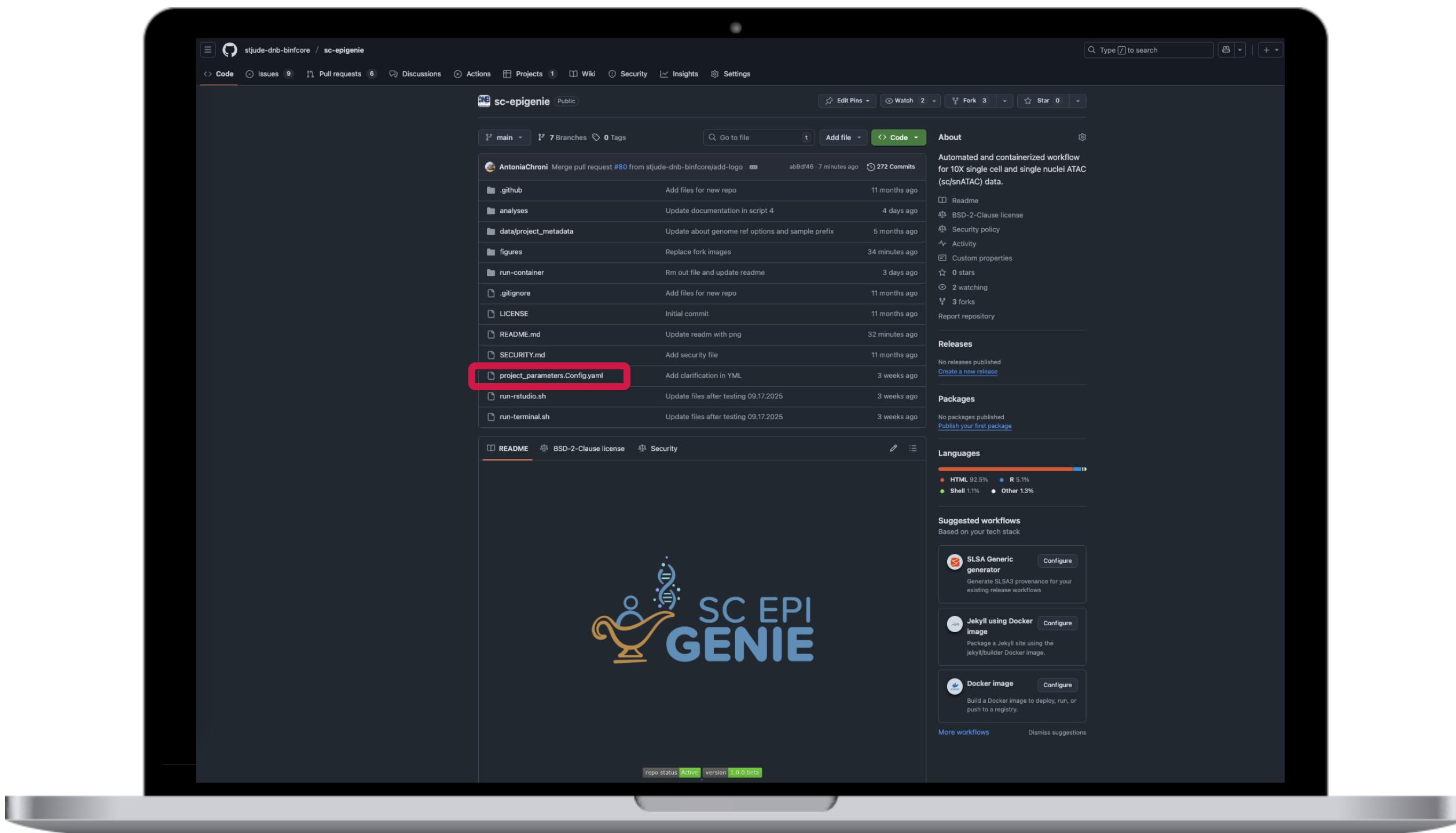




# My typical project organization system



# Customize your project based on the experiment!



# Project parameters file

```

project_parameters.Config.yaml
1 # the following parameters are the same across the project and might be needed in more than one module #
2 root_dir: "/sc-epigenie" # Absolute path to the main dir of the project where GitHub repo lives
3 data_dir: "/sc-epigenie/analyses/cellranger-analysis/results/02_cellranger_count/DefaultParameters" # Absolute path to data dir of the project with CellRanger output results. Options: "DefaultParameters", "For
4 metadata_dir: "/data/project_metadata" # Absolute path to metadata dir of the project.
5 metadata_file: "project_metadata.tsv" # Options: "project_metadata.tsv" (default) or name as user wants. It needs to be in 'tsv' format. It can include one or more samples, as long as it contains at least the
6 genome_name: "GRCm39" # define genome reference and versioning. Options: (1) human: "GRCh38" and "hg19"; and (2) mouse: "GRCm39", "mm10", and "mm9".
7 PROJECT_NAME: "Victoria_Knockout - Testing cohort"
8 PI_NAME: "Michael A. Dyer, PhD"
9 TASK_ID: "NA"
10 PROJECT_LEAD_NAME: "NA"
11 DEPARTMENT: "Developmental Neurobiology"
12 LEAD_ANALYSTS: "Antonia Chroni, PhD"
13 GROUP_LEAD: "Cody A. Ramirez, PhD"
14 CONTACT_EMAIL: "antonia.chroni@stjude.org"
15 PIPELINE: "Standard scATAC-Seq Analysis in 10X Genomics data"
16 START_DATE: "NA"
17 COMPLETION_DATE: "ONGOING"
18
19 # the following parameters are set up as default values and/or are specific for the following modules:
20 # `./analyses/fastqc-analysis`:
21 # FASTQ paths to the fastqc files with format: `path1/*R1-R3*.fastq.gz` are extracted from the `metadata_dir`.
22 # No need to manually define variables
23
24 # `./analyses/cellranger-analysis`:
25 genome_reference_path: "./reference_genomes/cellranger-sc-atac-seq/2020-A/mus_musculus/mm10/downloads/refdata-cellranger-arc-mm10-2020-A-2.0.0" # Absolute path to genome reference to be used for the `cellrange
26 cellranger_parameters: "DefaultParameters" # Options: "DefaultParameters", "ForcedCells8000Parameters", or else.
27 genome_name_cellranger: "GRCm39" #please define the genome of preference for dual genomes. In case for single genomes, please use the same as used for `genome_name` .
28 # Define the sample ID prefix(es) used in this project.
29 # Sample IDs should follow a format like: PREFIX001 (e.g., DYE001, ABC002).
30 # You can specify multiple prefixes if your project uses more than one.
31 sample_prefix:
32   - DYE
33   - ABC-
34   - XYZ_
35
36 # `./analyses/upstream-analysis`:
37 genome_name_upstream: "GRCm39" # Options: (1) human: "GRCh38" and "hg19"; and (2) mouse: "GRCm39", "mm10", and "mm9".
38 assay_signac_qc: "peaks" # Options: "peaks" (default ALWAYS)
39 # Genome-to-Ensembl Mapping Options: (1) human: "GRCh38" -> "EnsDb.Hsapiens.v86"(older); "EnsDb.Hsapiens.v98" | "hg19" -> "EnsDb.Hsapiens.v75"
40 #                                         (2) mouse: "GRCm39" -> "EnsDb.Mmusculus.v104" (or "EnsDb.Mmusculus.v105"), | "mm10" -> "EnsDb.Mmusculus.v79" and | "mm9" -> No official 'EnsDb' package, use "TxDb.Mmusculu
41 annotation_Ensembl_upstream: "EnsDb.Mmusculus.v104"
42 species_upstream: "Mus musculus" # Options: "Hsapiens"; "Mus musculus"
43 object_species_upstream: "AH95775" # Reference genome annotation ID. Options: (1) human: "GRCh38" -> "AH75011" | "AH14502"
44 #                                         (2) mouse: "GRCm39" -> AH95775"
45 min_cutoff_value_upstream: "q0" # Options: e.g., "q0" to use all peaks; "q75" to use the top 25% all peaks; "q95" to use the top 5% all peaks and so on.
46 #Count_peaks_min_upstream: "100" # Total counts across all peaks Optional; can correlate with fragments
47 #Count_peaks_max_upstream: "20000" # Total counts across all peaks Optional; can correlate with fragments
48 #Feature_peaks_min_upstream: "500" # Number of peaks with non-zero counts Optional
49 #Feature_peaks_max_upstream: "6000" # Number of peaks with non-zero counts Optional
50 #peak_region_fragments_min_upstream: "100" # Total fragments in called peak regions > 3000
51 #peak_region_fragments_max_upstream: "20000" "#20000"
52 pct_reads_in_peaks_value_upstream: "20" # % of reads in peaks (signal-to-noise) > 15 or > 20
53 blacklist_ratio_value_upstream: "0.05" # Fraction of reads in ENCODE blacklist regions (noise metric) < 0.05
54 #duplicate_value_upstream: "8000" # Duplication rate. Use only if computed
55 #mitochondrial_value_upstream: "5" # % of mitochondrial reads < 5
56 TSS_enrichment_value_upstream: "4" # Transcription start site enrichment score > 2
57 nucleosome_signal_value_upstream: "3" # Nucleosome periodicity signal < 4
58 use_threshold_filtering_upstream: "YES" # Options: "YES" (default) or "NO" (if user prefers to use percentile filtering)
59 condition_value1: "condition" # Use discrete values. Min #conditions per project: 1 and Max #conditions per project: 3. To use for visualization purposes and split UMAPs. Value to be extracted from column name in 'pr
60 condition_value2: NULL # Use discrete values. Min #conditions per project: 1 and Max #conditions per project: 3. To use for visualization purposes and split UMAPs. Value to be extracted from column name in 'pr
61 condition_value3: NULL # Use discrete values. Min #conditions per project: 1 and Max #conditions per project: 3. To use for visualization purposes and split UMAPs. Value to be extracted from column name in 'pr
62 print_pdf_seurat_multiple_samples: "YES" # Options: "YES" (default ALWAYS), for `01B_run_signac_qc_multiple_samples.R`
63 use_condition_split_seurat_multiple_samples: "NO" # Options: "NO" (default ALWAYS), for `01B_run_signac_qc_multiple_samples.R`
64 grouping: "orig.ident" # define grouping to use
65 use_condition_split_filter_object: "YES" # Options: "YES" (default) or "NO", for `03_run_filter_object.Rmd`
66 print_pdf_filter_object: "NO" # Options: "NO" (default ALWAYS), for `03_run_filter_object.Rmd`
67 use_scDblFinder_filtering_filter_object: "NO" # Options: "YES" or "NO" (default) , for `03_run_filter_object.Rmd` .
68 filename_filter_object_value: NULL # Options: NULL (default) or name if multiple methods are integrated and explored: "-without-scDblFinder" or "-with-scDblFinder"
69 #PCA_Feature_List_value: transcription.factor.gene.list # set for 04_run_filter_object.Rmd if necessary
70 filename_summary_report_value: NULL # Options: NULL (default) or name if multiple methods are integrated and explored: "-without-scDblFinder" or "-with-scDblFinder"
71 use_scDblFinder_filtering_summary_report: "NO" # Options: "YES" or "NO" (default) , for `05_run_summary_report.Rmd`.
72
73 # `./analyses/integrative-analysis`:
74 #     - without_scDblFinder: "YES" # Options: "YES" or "NO"

```



# How to run the code

The screenshot shows a GitHub repository's README page. A red box highlights the 'Running the Code' section, which contains instructions for configuring parameters and running shell scripts. Below this, another red box highlights the 'Sync Your Fork' section, which provides instructions for syncing with the main repository and lists useful git commands.

**Running the Code**

1. Configure Your Parameters

Replace the `project_parameters.Config.yaml` file with your own file paths and parameters.

2. Navigate to an Analysis Module

Change to the relevant directory and run the desired shell script:

```
cd ./sc-epigenie/analyses/<module_of_interest>
```

3. Sync Your Fork

User needs to ensure that the main branch of the forked repository is always up to date with `stjude-dnb-binfcore/sc-epigenie:main`.

If your fork is behind the main repository (`stjude-dnb-binfcore/sc-epigenie:main`), sync it to ensure you have the latest updates. This will update the main branch of your project repo with the new code and modules (if any). This will add code and not break any analyses already run in your project repo.

When syncing your forked repository with the main repository, please be cautious of any changes made to the following files, as they are typically modified and specified for project data analysis:

- `project_parameters.Config.yaml`

Before pulling the latest changes, stash any modifications you have made to these files. This ensures that you won't accidentally overwrite your changes when syncing with the main repository.

Some useful git commands:

```
git branch  
git checkout main  
git config pull.rebase false  
  
git status  
git add project_parameters.Config.yaml  
git commit -m "Update yaml"
```

Finally, `git pull` to get the most updated changes and code in your project repo. Please be mindful of any local changes in files in your project repo that you have done, e.g., `project_parameters.Config.yaml`. You will need to commit or stash (or restore) the changes to the yaml before completing the pull.

```
git pull
```



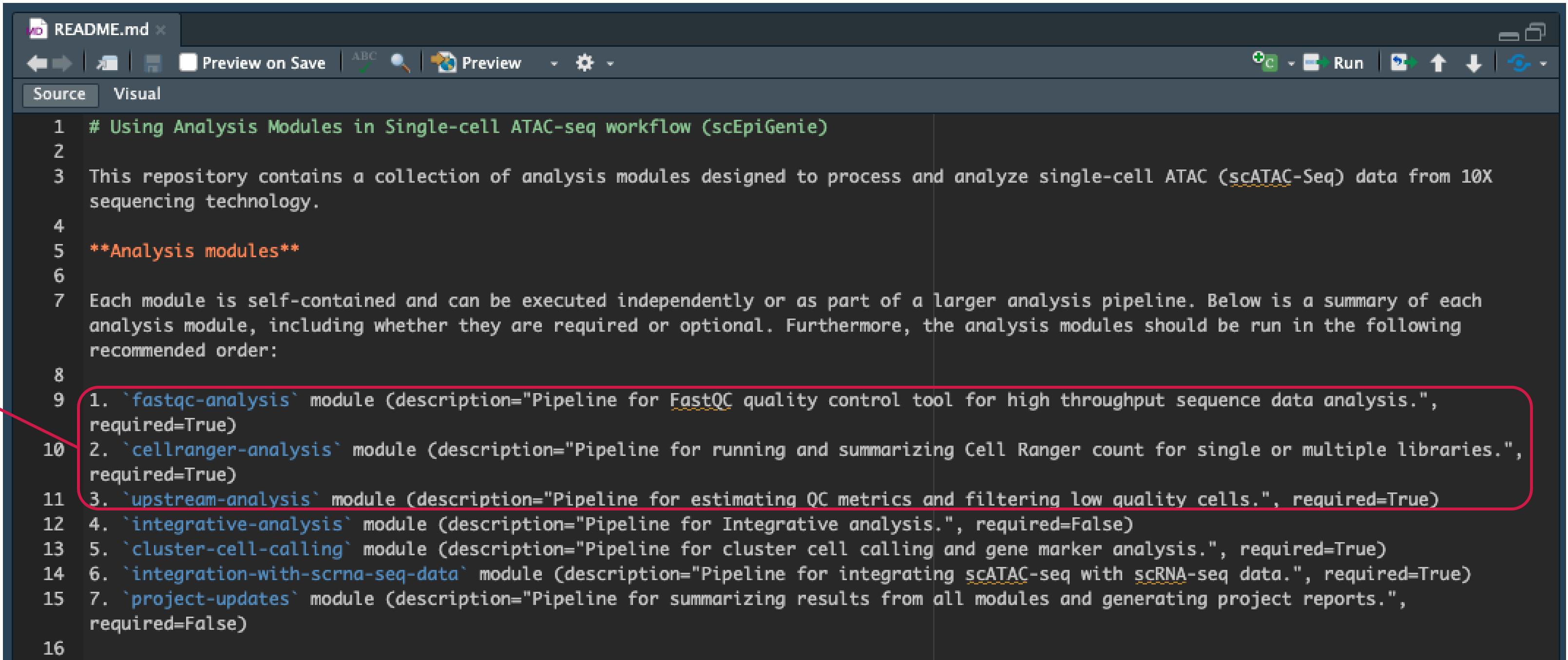
# scEpiGenie analysis modules

```
1 # Using Analysis Modules in Single-cell ATAC-seq workflow (scEpiGenie)
2
3 This repository contains a collection of analysis modules designed to process and analyze single-cell ATAC (scATAC-Seq) data from 10X sequencing technology.
4
5 **Analysis modules**
6
7 Each module is self-contained and can be executed independently or as part of a larger analysis pipeline. Below is a summary of each analysis module, including whether they are required or optional. Furthermore, the analysis modules should be run in the following recommended order:
8
9 1. `fastqc-analysis` module (description="Pipeline for FastQC quality control tool for high throughput sequence data analysis.", required=True)
10 2. `cellranger-analysis` module (description="Pipeline for running and summarizing Cell Ranger count for single or multiple libraries.", required=True)
11 3. `upstream-analysis` module (description="Pipeline for estimating QC metrics and filtering low quality cells.", required=True)
12 4. `integrative-analysis` module (description="Pipeline for Integrative analysis.", required=False)
13 5. `cluster-cell-calling` module (description="Pipeline for cluster cell calling and gene marker analysis.", required=True)
14 6. `integration-with-scrna-seq-data` module (description="Pipeline for integrating scATAC-seq with scRNA-seq data.", required=True)
15 7. `project-updates` module (description="Pipeline for summarizing results from all modules and generating project reports.", required=False)
16
```



# scEpiGenie analysis modules

Upstream

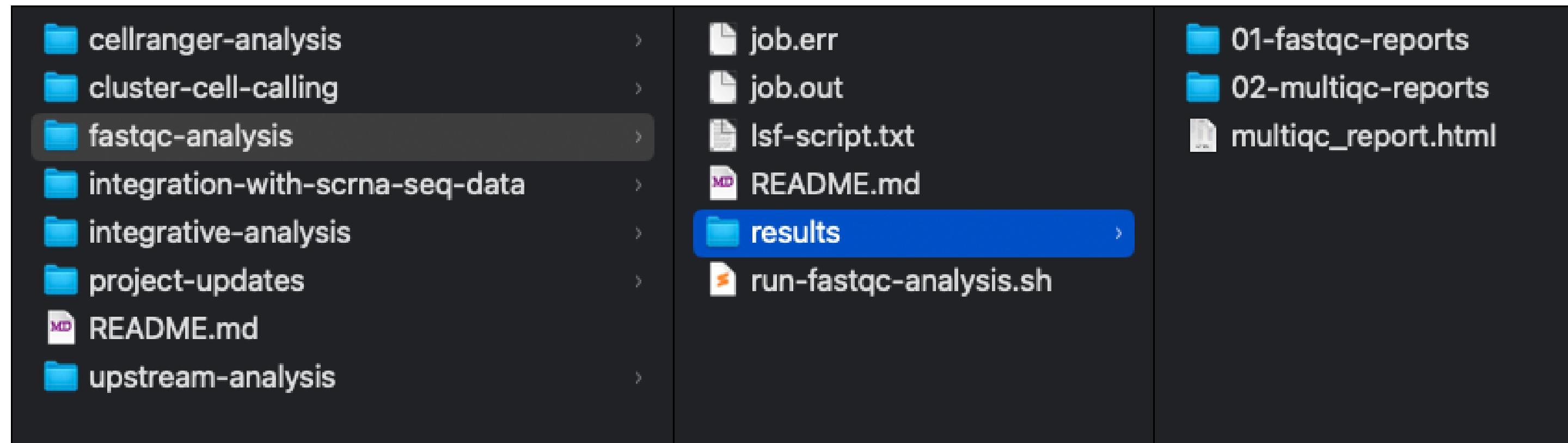


```
1 # Using Analysis Modules in Single-cell ATAC-seq workflow (scEpiGenie)
2
3 This repository contains a collection of analysis modules designed to process and analyze single-cell ATAC (scATAC-Seq) data from 10X sequencing technology.
4
5 **Analysis modules**
6
7 Each module is self-contained and can be executed independently or as part of a larger analysis pipeline. Below is a summary of each analysis module, including whether they are required or optional. Furthermore, the analysis modules should be run in the following recommended order:
8
9 1. `fastqc-analysis` module (description="Pipeline for FastQC quality control tool for high throughput sequence data analysis.", required=True)
10 2. `cellranger-analysis` module (description="Pipeline for running and summarizing Cell Ranger count for single or multiple libraries.", required=True)
11 3. `upstream-analysis` module (description="Pipeline for estimating QC metrics and filtering low quality cells.", required=True)
12 4. `integrative-analysis` module (description="Pipeline for Integrative analysis.", required=False)
13 5. `cluster-cell-calling` module (description="Pipeline for cluster cell calling and gene marker analysis.", required=True)
14 6. `integration-with-scrna-seq-data` module (description="Pipeline for integrating scATAC-seq with scRNA-seq data.", required=True)
15 7. `project-updates` module (description="Pipeline for summarizing results from all modules and generating project reports.", required=False)
16
```



# `fastqc-analysis` module

description="Pipeline for FastQC quality control tool for high throughput sequence data analysis.", required=True



# `run-fastqc-analysis.sh`

---



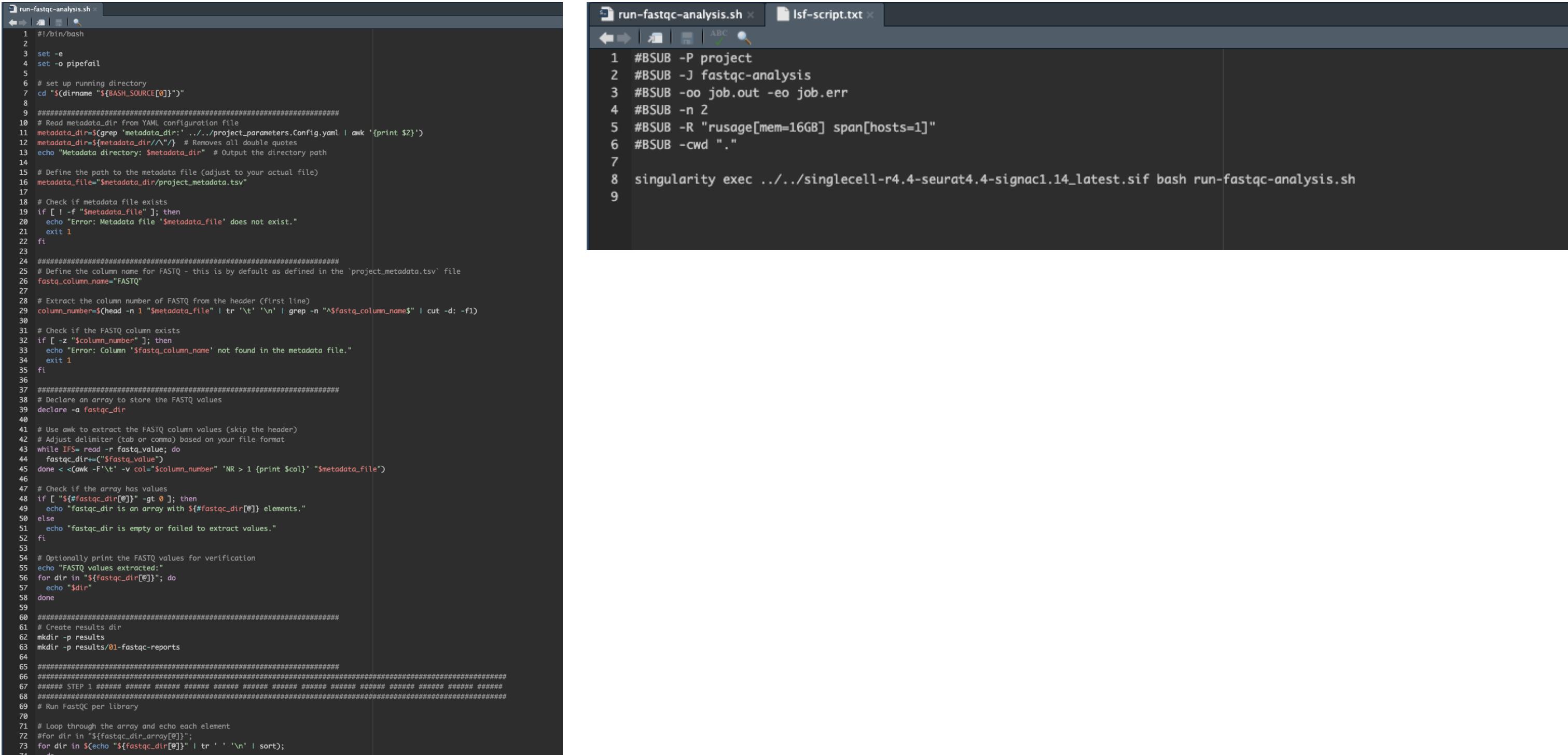
```

run-fastqc-analysis.sh
#!/bin/bash
set -e
set -o pipefail
# set up running directory
cd "$(dirname "${BASH_SOURCE[0]}")"
#####
# Read metadata_dir from YAML configuration file
metadata_dir=$(grep 'metadata_dir:' ../../project_parameters.Config.yaml | awk '{print $2}')
metadata_dir=${metadata_dir//"/"} # Removes all double quotes
echo "Metadata directory: $metadata_dir" # Output the directory path
# Define the path to the metadata file (adjust to your actual file)
metadata_file="$metadata_dir/project_metadata.tsv"
#####
# Check if metadata file exists
if [ ! -f "$metadata_file" ]; then
    echo "Error: Metadata file '$metadata_file' does not exist."
    exit 1
fi
#####
# Define the column name for FASTQ - this is by default as defined in the `project_metadata.tsv` file
fastq_column_name="FASTQ"
#####
# Extract the column number of FASTQ from the header (first line)
column_number=$(head -n 1 "$metadata_file" | tr '\t' '\n' | grep -n "^$fastq_column_name$" | cut -d: -f1)
#####
# Check if the FASTQ column exists
if [ -z "$column_number" ]; then
    echo "Error: Column '$fastq_column_name' not found in the metadata file."
    exit 1
fi
#####
# Declare an array to store the FASTQ values
declare -a fastqc_dir
#####
# Use awk to extract the FASTQ column values (skip the header)
# Adjust delimiter (tab or comma) based on your file format
while IFS= read -r fastq_value; do
    fastqc_dir+=("$fastq_value")
done < <(awk -F'\t' -v col="$column_number" 'NR > 1 {print $col}' "$metadata_file")
#####
# Check if the array has values
if [ ${#fastqc_dir[@]} -gt 0 ]; then
    echo "fastqc_dir is an array with ${#fastqc_dir[@]} elements."
else
    echo "fastqc_dir is empty or failed to extract values."
fi
#####
# Optionally print the FASTQ values for verification
echo "FASTQ values extracted:"
for dir in ${fastqc_dir[@]}; do
    echo "$dir"
done
#####
# Create results dir
mkdir -p results
mkdir -p results/01-fastqc-reports
#####
##### STEP 1 #####
#####
# Run FastQC per library
#####
# Loop through the array and echo each element
#for dir in ${fastqc_dir_array[@]};
for dir in $(echo "${fastqc_dir[@]}" | tr ' ' '\n' | sort);
do
    echo
done

```



# `run-fastqc-analysis.sh` and `lsf-script.txt`



The image shows a terminal window with two tabs open. The left tab contains the script `run-fastqc-analysis.sh`, and the right tab contains the file `lsf-script.txt`. The `run-fastqc-analysis.sh` script is a bash script that performs several steps: it sets up the running directory, reads metadata from a YAML file, checks for a metadata file, defines the path to the metadata file, extracts the FASTQ column number, checks if the FASTQ column exists, declares an array to store FASTQ values, uses awk to extract the FASTQ column values, checks if the array has values, optionally prints the FASTQ values for verification, creates results directories, and runs FastQC per library. The `lsf-script.txt` file is an LSF submission script that submits a job named `fastqc-analysis` with the following parameters: project, job output file (job.out), error output file (job.err), number of threads (n=2), resource usage (mem=16GB, span[hosts=1]), current working directory (cwd ".") and singularity exec command to run the `run-fastqc-analysis.sh` script.

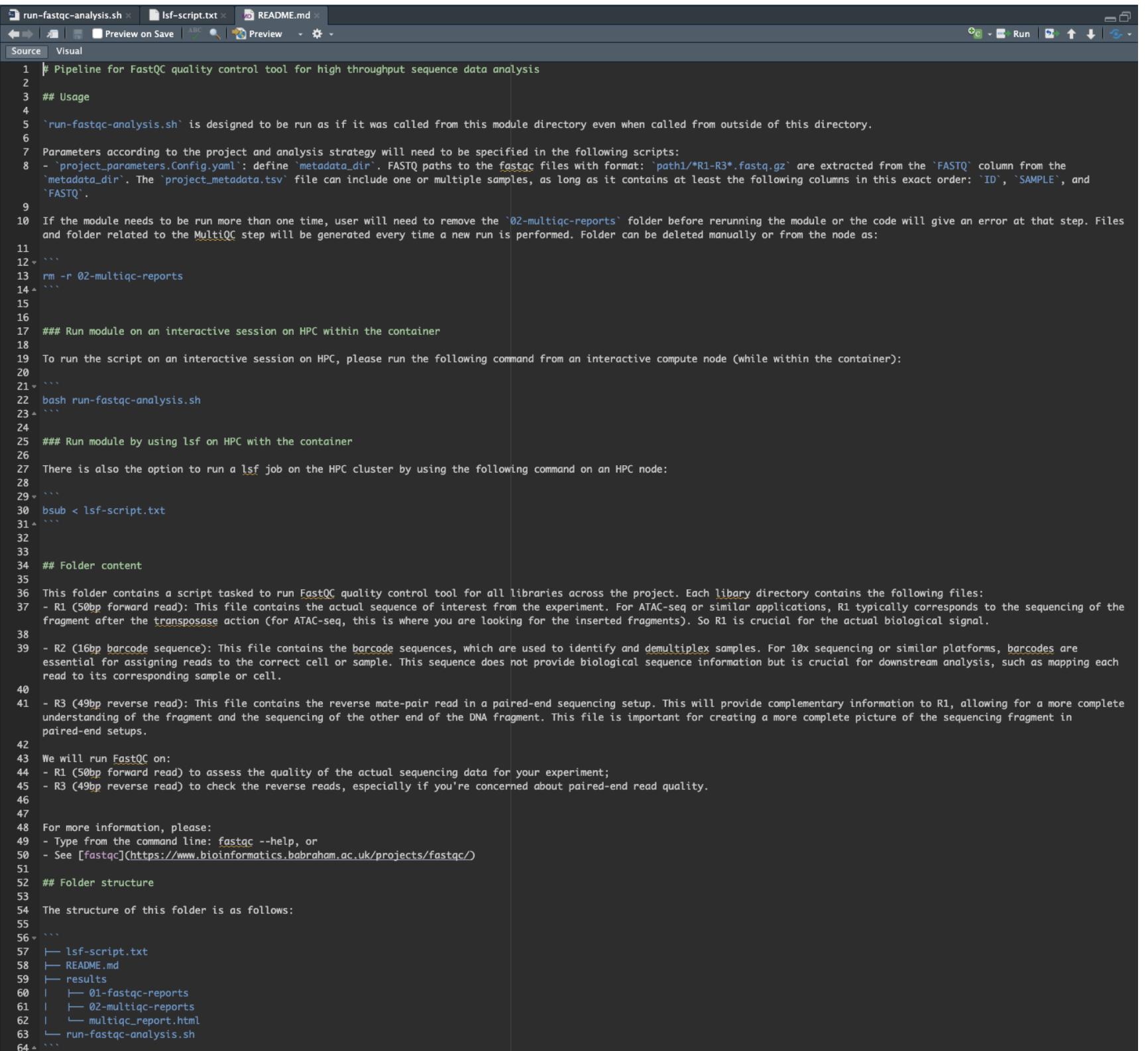
```

# run-fastqc-analysis.sh
#!/bin/bash
set -e
set -o pipefail
# set up running directory
cd "${dirname "${BASH_SOURCE[0]}"}"
#####
# Read metadata_dir from YAML configuration file
metadata_dir=$(grep 'metadata_dir:' ../../project_parameters.Config.yaml | awk '{print $2}')
metadata_dir=${metadata_dir//"/"} # Removes all double quotes
echo "Metadata directory: $metadata_dir" # Output the directory path
#####
# Define the path to the metadata file (adjust to your actual file)
metadata_file="$metadata_dir/project_metadata.tsv"
#####
# Check if metadata file exists
if [ ! -f "$metadata_file" ]; then
    echo "Error: Metadata file '$metadata_file' does not exist."
    exit 1
fi
#####
# Define the column name for FASTQ - this is by default as defined in the `project_metadata.tsv` file
fastq_column_name="FASTQ"
#####
# Extract the column number of FASTQ from the header (first line)
column_number=$(head -n 1 "$metadata_file" | tr '\t' '\n' | grep -n "^$fastq_column_name$" | cut -d: -f1)
#####
# Check if the FASTQ column exists
if [ -z "$column_number" ]; then
    echo "Error: Column '$fastq_column_name' not found in the metadata file."
    exit 1
fi
#####
# Declare an array to store the FASTQ values
declare -a fastqc_dir
#####
# Use awk to extract the FASTQ column values (skip the header)
# Adjust delimiter (tab or comma) based on your file format
IFS= read -r fastq_value; do
    fastqc_dir+=("$fastq_value")
done < <(awk -F'\t' -v col="$column_number" 'NR > 1 {print $col}' "$metadata_file")
#####
# Check if the array has values
if [ ${#fastqc_dir[@]} -gt 0 ]; then
    echo "fastqc_dir is an array with ${#fastqc_dir[@]} elements."
else
    echo "fastqc_dir is empty or failed to extract values."
fi
#####
# Optionally print the FASTQ values for verification
echo "FASTQ values extracted:"
for dir in ${fastqc_dir[@]}; do
    echo "$dir"
done
#####
# Create results dir
mkdir -p results
mkdir -p results/01-fastqc-reports
#####
##### STEP 1 #####
#####
# Run FastQC per library
#####
# Loop through the array and echo each element
#for dir in ${fastqc_dir_array[@]};
#do
#    echo "${fastqc_dir[@]}" | tr ' ' '\n' | sort;
#done

```



# README.md



```

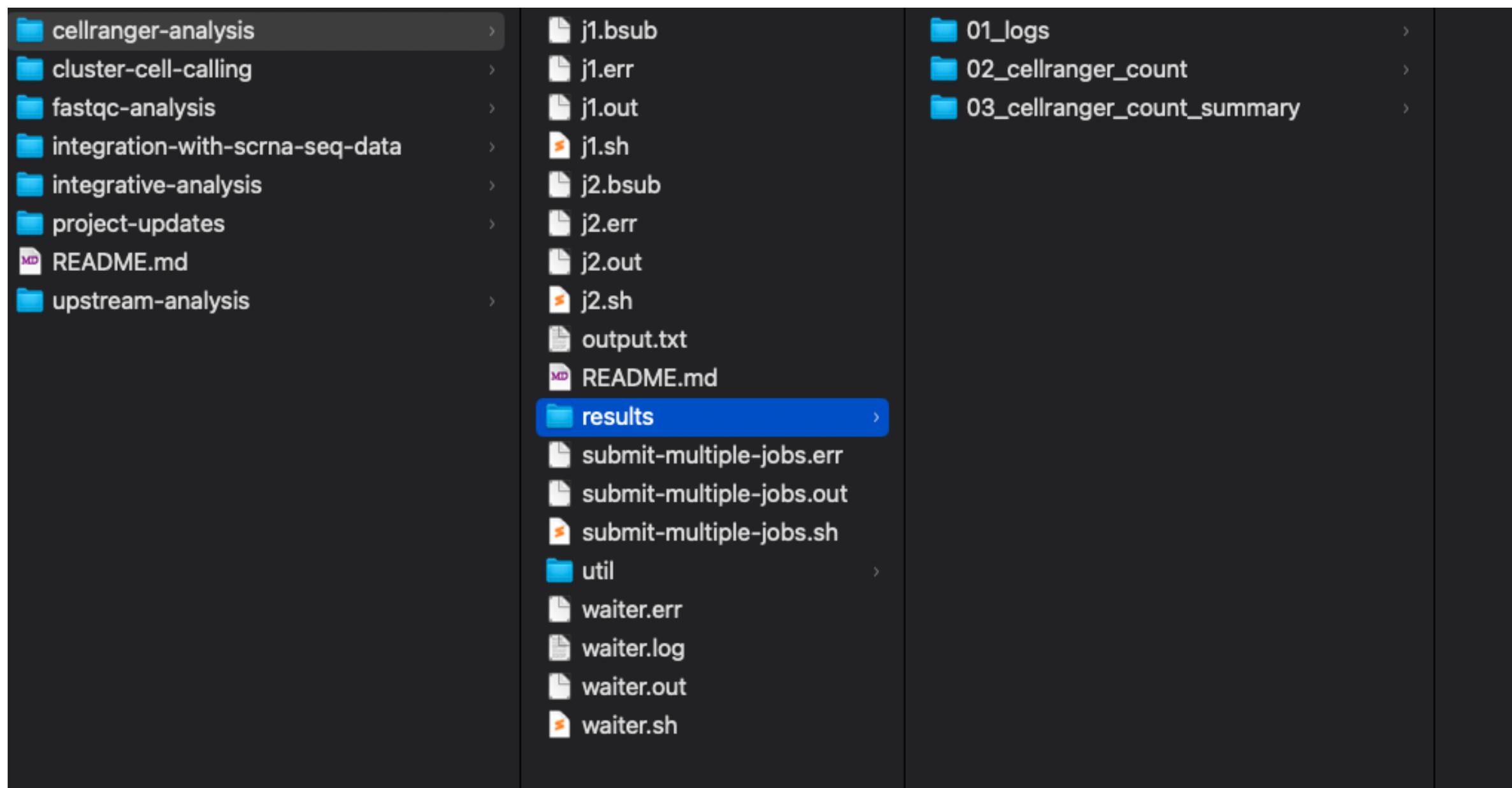
1  # Pipeline for FastQC quality control tool for high throughput sequence data analysis
2
3  ## Usage
4
5  `run-fastqc-analysis.sh` is designed to be run as if it was called from this module directory even when called from outside of this directory.
6
7  Parameters according to the project and analysis strategy will need to be specified in the following scripts:
8  - `project_parameters.Config.yaml`: define `metadata_dir`. FASTQ paths to the fastqc files with format: `path1/*R1-R3*.fastq.gz` are extracted from the `FASTQ` column from the `metadata_dir`. The `project_metadata.tsv` file can include one or multiple samples, as long as it contains at least the following columns in this exact order: `ID`, `SAMPLE`, and `FASTQ`.
9
10 If the module needs to be run more than one time, user will need to remove the `02-multiqc-reports` folder before rerunning the module or the code will give an error at that step. Files and folder related to the MultiQC step will be generated every time a new run is performed. Folder can be deleted manually or from the node as:
11
12 `````
13 rm -r 02-multiqc-reports
14 `````
15
16
17 ### Run module on an interactive session on HPC within the container
18
19 To run the script on an interactive session on HPC, please run the following command from an interactive compute node (while within the container):
20
21 `````
22 bash run-fastqc-analysis.sh
23 `````
24
25 ### Run module by using lsf on HPC with the container
26
27 There is also the option to run a lsf job on the HPC cluster by using the following command on an HPC node:
28
29 `````
30 bsub < lsf-script.txt
31 `````
32
33
34 ## Folder content
35
36 This folder contains a script tasked to run FastQC quality control tool for all libraries across the project. Each library directory contains the following files:
37 - R1 (50bp forward read): This file contains the actual sequence of interest from the experiment. For ATAC-seq or similar applications, R1 typically corresponds to the sequencing of the fragment after the transposase action (for ATAC-seq, this is where you are looking for the inserted fragments). So R1 is crucial for the actual biological signal.
38
39 - R2 (16bp barcode sequence): This file contains the barcode sequences, which are used to identify and demultiplex samples. For 10x sequencing or similar platforms, barcodes are essential for assigning reads to the correct cell or sample. This sequence does not provide biological sequence information but is crucial for downstream analysis, such as mapping each read to its corresponding sample or cell.
40
41 - R3 (49bp reverse read): This file contains the reverse mate-pair read in a paired-end sequencing setup. This will provide complementary information to R1, allowing for a more complete understanding of the fragment and the sequencing of the other end of the DNA fragment. This file is important for creating a more complete picture of the sequencing fragment in paired-end setups.
42
43 We will run FastQC on:
44 - R1 (50bp forward read) to assess the quality of the actual sequencing data for your experiment;
45 - R3 (49bp reverse read) to check the reverse reads, especially if you're concerned about paired-end read quality.
46
47
48 For more information, please:
49 - Type from the command line: fastqc --help, or
50 - See [fastqc](https://www.bioinformatics.babraham.ac.uk/projects/fastqc/)
51
52 ## Folder structure
53
54 The structure of this folder is as follows:
55
56 `````
57 |   lsf-script.txt
58 |   README.md
59 |   results
60 |   |   01-fastqc-reports
61 |   |   02-multiqc-reports
62 |   |   multiqc_report.html
63 |   |   run-fastqc-analysis.sh
64 `````

```

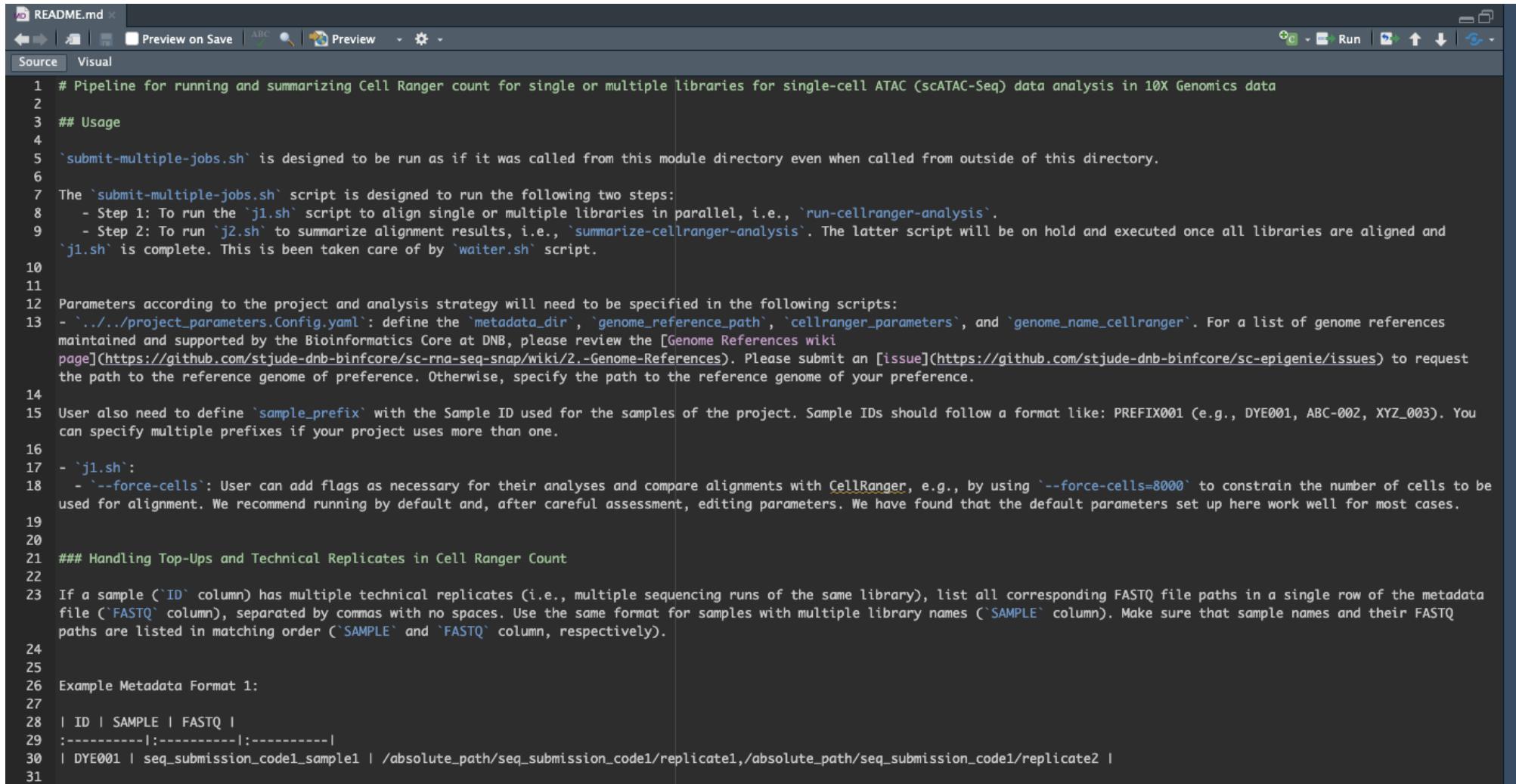


# `cellranger-analysis` module

description="Pipeline for running and summarizing Cell Ranger count for single or multiple libraries.", required=True.



# `cellranger-analysis` module



```

32 Example Metadata Format 2:
33 | ID | SAMPLE | FASTQ |
34 |-----|:-----|:-----|
35 | DYE001 | seq_submission_code1_sample1,seq_submission_code2_sample2 | /absolute_path/seq_submission_code1,/absolute_path/seq_submission_code2 |
36
37
38
39 Cell Ranger will automatically recognize these as multiple libraries for the same sample ('ID' column) and merge them into a single output for the sample during processing. There is no need to manually combine or rename the files—simply format them correctly in the metadata file, and the pipeline will handle the rest.
40
41
42 #### Run module on HPC
43
44 To run all of the scripts in this module sequentially on an interactive session on HPC, please run the following command from an interactive compute node:
45
46 ...
47 bsub < submit-multiple-jobs.sh
48 ...
49
50 Please note that this will run the analysis module outside of the container while submitting lsf job on HPC. This is currently the only option of running the 'cellranger-analysis' module. By default, we are using 'python/3.9.9' and 'cellranger-atac/2.1.0' as available on St Jude HPC.
51
52
53 ## Folder content
54
55 This folder contains scripts tasked to run and summarize Cell Ranger count for single or multiple libraries for single-cell ATAC (scATAC-Seq) data analysis in 10X Genomics data across the project. For more information and updates, please see [Cell Ranger support page](https://support.10xgenomics.com/single-cell-atac/software/pipelines/latest/using/count).
56
57 This module uses CellRanger-atac v2.1.0 for the alignment.
58
59 For more information, on how to review the web summary file in the output folder of the Cell Ranger ATAC analysis software, please see the [Interpreting Cell Ranger ATAC Web Summary Files for Single Cell ATAC Assay](https://www.10xgenomics.com/support/epi-atac/documentation/steps/sequencing/interpreting-cell-ranger-atac-web-summary-files-for-single-cell-atac-assay) file.
60
61
62 ## Folder structure
63
64 The structure of this folder is as follows:
65 ...
66
67 ├── j1.sh
68 ├── j2.sh
69 └── README.md
70 └── results
71   └── 01_logs
72   └── 02_cellranger_count
73   |   └── ${cellranger_parameters}
74   |       └── 03_cellranger_count_summary
75   |           └── ${cellranger_parameters}
76   └── submit-multiple-jobs.sh
77 └── util
78   └── summarize_cellranger_results.py
79 └── ...
80 ...
81

```

The screenshot shows a code editor window with the file 'README.md' open. The code is a README for the 'cellranger-analysis' module. It includes sections for pipeline usage, parameters, and folder structure. A vertical line on the left indicates the current cursor position.

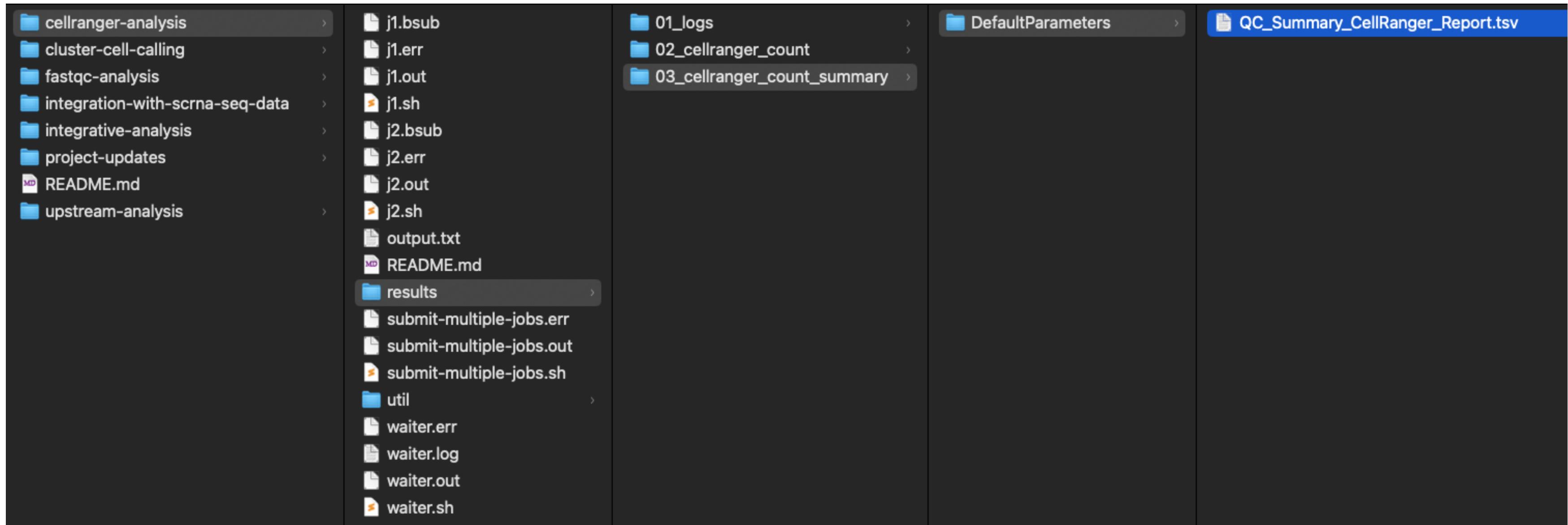


# `cellranger-analysis` module

<b>cellranger-analysis</b>	j1bsub j1.err j1.out j1.sh j2bsub j2.err j2.out j2.sh output.txt README.md results submit-multiple-jobs.err submit-multiple-jobs.out submit-multiple-jobs.sh util waiter.err waiter.log waiter.out waiter.sh	01_logs 02_cellranger_count 03_cellranger_summary	DefaultParameters	D 9 D 0 D 1 D 2	cmdline filelist finalstate invocation jobmode log mrosource perf sitecheck tags timestamp uuid vdrkill versions 9.mri.tgz outs SC_ATAC_COUNTER_CS
----------------------------	--	---	-------------------	--------------------------	--



# `cellranger-analysis` module



# `cellranger-analysis` module

---

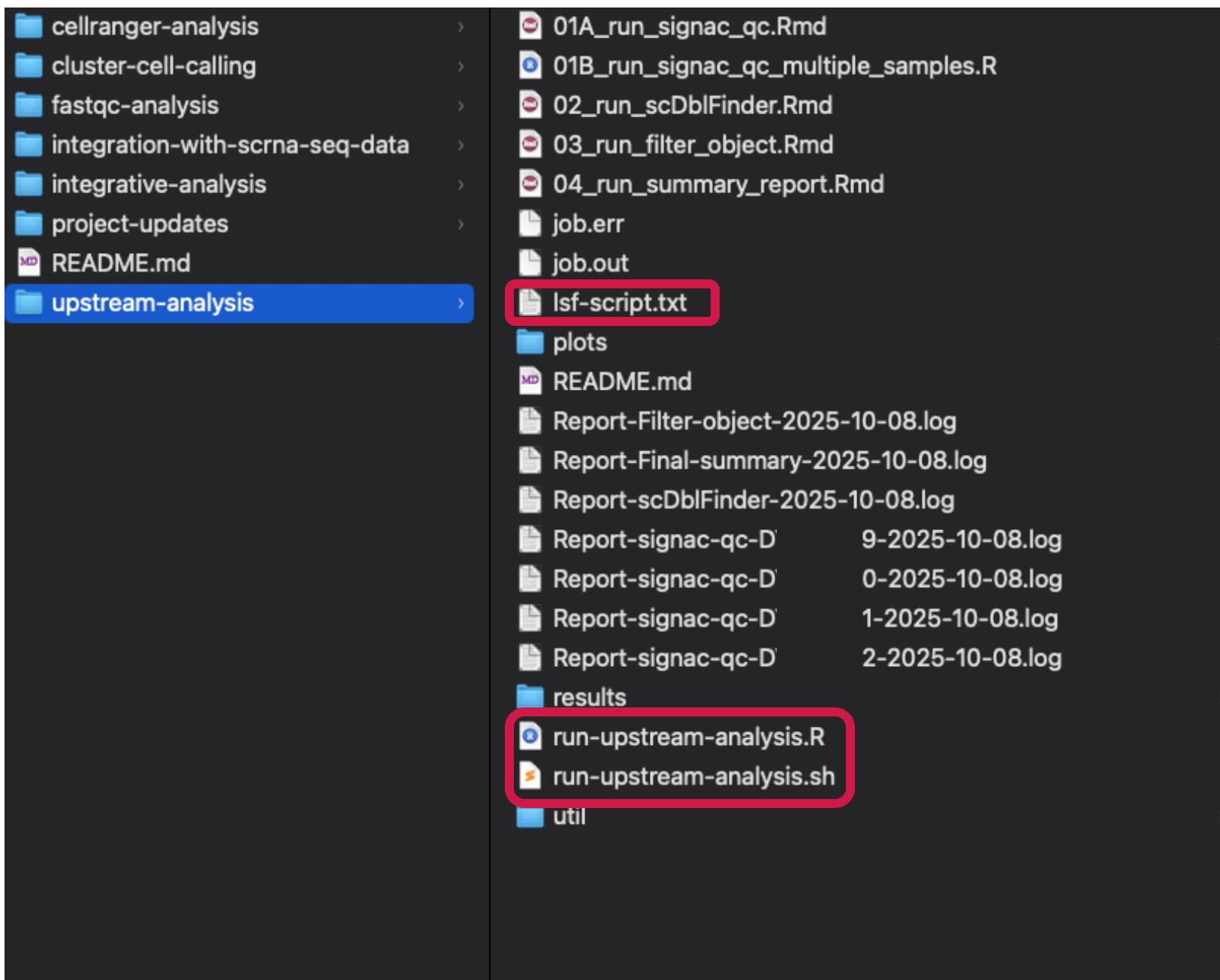


- **sc-atac-bed-builder.** 🚧 Currently under construction. Stay tuned! 🚧
  - <https://github.com/stjude-dnb-binfcore/sc-atac-seq-bed-builder>



# `upstream-analysis` module

description="Pipeline for estimating QC metrics and filtering low quality cells.", required=True.



# ‘upstream-analysis’ module

```

README.md x
Preview on Save ABC Preview Run
Source Visual
1 # Pipeline for estimating QC metrics for sc-ATAC-Seq Analysis in 10X Genomics data
2
3 ## Usage
4
5 `run-upstream-analysis.sh` is designed to be run as if it was called from this module directory even when called from outside of this directory.
6 Parameters according to the project and analysis strategy will need to be specified in the following scripts:
7 - `project_parameters.Config.yaml` located at the `root_dir`.
8 - `future.globals.maxSize` is hardwired coded in the `run-upstream-analysis.R`. If necessary, user can increase/decrease resources.
9
10 ### Run module on an interactive session on HPC within the container
11 To run all of the Rscripts in this module sequentially on an interactive session on HPC, please run the following command from an interactive compute node (while within the container):
12 ...
13 bash run-upstream-analysis.sh
14 ...
15 ...
16 ...
17 ...
18 ...
19 ...
20 ...
21 ...
22 There is also the option to run a lsf job on the HPC cluster by using the following command on an HPC node:
23 ...
24 ...
25 bsub < lsf-script.txt
26 ...
27 ...
28 ...
29 ...
30 ...
31 This folder contains scripts tasked to:
32 (1) Infer QC metrics and associated plots to visually explore the quality of each library of the project.
33 (2) Evaluate QC metrics and set filters to remove low quality cells in 10x single-cell-ATAC-sequencing libraries.
34 ...
35 ...
36 ...
37 The pipeline offers flexibility for users to include or exclude methods and adjust the workflow during QC based on factors such as sequence type, expected cell number, experiment type, and genome reference.
38 By default, the pipeline runs all methods from steps (1-2). Step (1) is mandatory for basic QC filtering, while integration of step (2) is optional. This can be configured in the `project_parameters.Config.yaml` file. However, we recommend reviewing all results, as this can provide valuable insights into the overall quality of each library.
39 ...
40 ...
41 ...
42 ...
43 ...
44 [Signac](https://stuartlab.org/signac/articles/pbmc_vignette) workflow is implemented to pre-process, filter and plot the ATAC-sequencing data. For more tutorials, see [Introduction to single cell ATAC data analysis in R](https://www.youtube.com/watch?v=e2396GKMRY&ab_channel=Sanbomics) and [How to analyze single-cell ATAC-Seq data in R | Detailed Signac Workflow Tutorial](https://www.youtube.com/watch?v=yEKZJYjc5DY&ab_channel=Bioinformagician).
45 ...
46 The CellRanger output from the `cellranger-analysis` module will be used for this step. User will have to define `params` as needed for their experiment.
47 - Calculate QC metrics: Nucleosome banding pattern, Transcriptional start site (TSS) enrichment score, Total number of fragments in peaks, Fraction of fragments in peaks, Ratio reads in genomic blacklist regions, etc.
48 - Before and after filter: Plot "pct_reads_in_peaks", "peak_region.fragments", "TSS.enrichment", "blacklist_ratio", "nucleosome.signal", "nCount_peaks", "nFeature_peaks", "pct_reads_in_peaks_promoters", "pct_reads_in_peaks_enancers", "duplicate", "mitochondrial".
49 - Data are normalized by using term frequency-inverse document frequency (TF-IDF) normalization. Then, we select the top n% of features (peaks) for dimensional reduction, or remove features present in less than n cells with the [FindTopFeatures()](https://stuartlab.org/signac/reference/findtopfeatures) function. We next run singular value decomposition (SVD) on the TD-IDF matrix, using the features (peaks) selected above. This returns a reduced dimension representation of the object (for users who are more familiar with scRNA-seq, you can think of this as analogous to the output of PCA). The combined steps of TF-IDF followed by SVD are known as latent semantic indexing (LSI). After that the cells are embedded in a low-dimensional space we can run UMAP from the Seurat package.
50 ...
51 Here, the user can select to implement the following strategies to remove low quality cells via the `run_QC` function:
52 - `approach 1` Filter cells based on various variables as defined in the `params` .
53 - `approach 2` Use percentile filtering. This will be defined in the `params` by setting `use_threshold_filtering_upstream: "NO"`.
54 ...
55 Moreover, only libraries with more than 500 cells will be kept for merging and integration purposes. This value is a commonly used threshold for many single-cell ATAC-seq studies as a minimum for obtaining reliable and meaningful analysis.
56 - Statistical Power: At least 500 cells are typically needed to ensure the analysis has enough statistical power to detect meaningful biological signals.
57 - Cell Diversity: With fewer cells, you may not capture sufficient cellular diversity, leading to incomplete or biased results.
58 - Clustering: Some clustering algorithms in single-cell RNA-seq require a minimum number of cells to create robust, meaningful clusters.
59 ...

```

60 **## Genome-to-Ensembl Mapping Options**  
61  
62 We need to assign gene annotations from Ensembl based on the genome reference used for the cohort. For more information on the various Ensembl releases, you can refer to the [Table of Assemblies](https://useast.ensembl.org/info/website/archives/assembly.html) and [Build Notes for Reference Packages](https://www.10xgenomics.com/support/software/cell-ranger/downloads/cr-ref-build-steps#ref-2020-a).

63 | Species | Genome Build | Ensembl Version(s) | Annotation Package |  
64 | --- | --- | --- | --- |  
65 | Human | hg19 (GRCh37) | v75 | Ensembl.Hsapiens.v75' |  
66 | Human | hg38 (GRCh38) | v86 (older), v98 (newer) | Ensembl.Hsapiens.v86' |  
67 | Mouse | mm10 (GRCm38) | - | \*No official\* Ensembl ; use `TxDb.Musculus.UCSC.mm9.knownGene` |  
68 | Mouse | mm12 (GRCm39) | v79 | Ensembl.Musculus.v79' |  
69 | Mouse | mm13 (GRCm39) | v104, v105 | Ensembl.Musculus.v104' | Ensembl.Musculus.v105' |  
70 |  
71  
72 **## Key QC Metrics and Suggested Thresholds (Per Cell/Nucleus)**  
73 | \*\*Metric\*\* | \*\*Interpretation\*\* | \*\*Low-Quality Threshold\*\* | \*\*High-Quality Range\*\* |  
74 | --- | --- | --- | --- |  
75 | \*\*Fragments in peaks\*\* | Proportion of reads in peaks – reflects signal-to-noise | < 20-30% | > 30-50% |  
76 | \*\*TSS enrichment score\*\* | Enrichment of reads around transcription start sites | < 4-6 | > 6-10 |  
77 | \*\*Nucleosome signal\*\* | Ratio of mono- to di-/tri-nucleosome fragments (chromatin state) | > 4 | < 2-3 |  
78 | \*\*Total fragments\*\* | Total number of fragments per cell. Avoid very low or very high. | < 1k or > 50k | 3k-30k |  
79 | \*\*Blacklist ratio\*\* | Fraction of reads in ENCODE blacklist regions (noise = High = poor quality) | > 0.01 | < 0.01 |  
80 | \*\*Mono-/multi-nucleosome\*\* | Fragment length distribution should show clear banding | Poor/no banding | Visible nucleosome bands |  
81 | \*\*Doublet score\*\* | Estimate of multiplets (2+ cells) per barcode (Use to exclude multiplets) | High | Low (filtered out) |  
82 |  
83  
84  
85 **## Quality control**  
86  
87 [Baek and Lee, 2020](https://www.sciencedirect.com/science/article/pii/S2001037020303019#s0015) emphasized the importance of filtering out barcodes corresponding to low-quality cells or doublets after processing sequencing read data. In general, single-cell sequencing QC relies on metrics like read counts (count depth) and feature counts per barcode ([Luecken and Theis, 2019](https://www.scopus.com/record/display.uri?eid=2-s2.0-85067863532&origin=inward&tqid=44862f3571/b260b21d41030ddd82fe)). Barcodes with either abnormally low or high read/feature counts are typically flagged as low-quality cells or multiplets, respectively.

88 However, scATAC-seq data offers additional QC metrics that better reflect chromatin accessibility quality. Commonly used indicators include the fraction of reads in peaks (FRIP), promoter read ratios, blacklist region read ratios, and transcription start site (TSS) enrichment scores ([Buenrostro et al., 2015](https://www.nature.com/articles/nature14590), [Fang et al., 2021](https://www.nature.com/articles/s41467-021-21583-9), [Granaia et al., 2021](https://www.nature.com/articles/s41588-021-00790-6)). Barcodes lacking characteristic nucleosome banding patterns—typical of high-quality ATAC-seq data—are also excluded ([Cusanovich et al., 2015](https://www.science.org/doi/10.1126/science.aab1601src-getfrt&utm\_source=sciedirect\_contenthosting&getfrt\_integrator=sciedirect\_contenthosting), [Cusanovich et al., 2018](https://www.cell.com/cell/fulltext/S0092-8674(18)30855-9)). Additionally, peaks located in blacklist regions or overlapping housekeeping genes may be removed during feature filtering ([Fang et al., 2021](https://www.nature.com/articles/s41467-021-21583-9)).

89 It's important to note that no universal QC thresholds apply to all datasets. QC criteria should be adapted based on specific sample characteristics, such as data complexity, cellular heterogeneity, expected cell types, batch effects, or sequencing platform used ([Baek and Lee, 2020](https://www.sciencedirect.com/science/article/pii/S2001037020303019#s0015)).

90 For more information on QC and other sc-ATAC-Seq related methods, see [Lei Xiong et al., 2019](https://www.nature.com/articles/s41467-019-12630-7#Sec10), [Li et al., 2021](https://www.nature.com/articles/s41467-021-26530-2), [Stuart et al., 2021](https://www.nature.com/articles/s41592-021-01282-5#Sec9), [Taavitsainen et al., 2021](https://www.nature.com/articles/s41467-021-25624-1#Sec10), [De Kop et al., 2023](https://www.nature.com/articles/s41587-023-01881-x), [Camache et al., 2023](https://link.springer.com/article/10.1186/s13578-023-01120-5#Sec10)

91  
92 We recommend that the user use the following parameters for initial 'scATAC' QC, and then adjust accordingly if necessary:

Parameter	Suggested Value	Corrected Comment
peak_region.fragments_min_upstream	~100	Cells with very few fragments in peaks likely represent background noise, empty droplets, or low-quality nuclei.
pct_reads_in_peaks_value_upstream	~15%	% reads in peaks => 15% (or > 20%)
blacklist_ratio_value_upstream	~0.05%	% reads in ENCODE blacklist - << 0.05%
mitochondrial_value_upstream	~5%	% mitochondrial reads - << 5%
TSS.enrichment_value_upstream	~10%	TSS enrichment score - => 2%
nucleosome_signal_value_upstream	~4%	Nucleosome signal - << 4%

93  
94 **(2) Estimating and filtering out doublets**  
95 Popular approach of single-cell uses oil droplets or wells to isolate single cells along with barcoded beads. Depending on the cell density loaded, a proportion of reaction volumes (i.e. droplets or wells) will capture more than one cell, forming 'doublets' (or 'multiplets'), i.e. two or more cells captured by a single reaction volume and thus sequenced as a single-cell artifact.

96 The proportion of doublets is proportional to the number of cells captured. It is common in single-cell experiments to have 10-20% doublets, making accurate doublet detection critical.

97 Doublets are prevalent in single-cell sequencing data and can lead to artificial findings. We will use a computational approach to calculate and remove doublets from the library. Here, we use [ScDbFinder](https://bioconductor.org/packages-devel/bioc/vignettes/scDbFinder/inst/doc/scDbFinder.html) method for identifying doublets/multiplets in single-cell data.

98 The 'seurat\_obj\_raw.rds' object from step (1) is used for this step.  
- Summary table with doublet metrics and doublets prediction plot are generated.

99  
**(3) Merging filtered data**  
100 Next, we merge count matrices from steps (1-2) after filtering out low quality cells and doublets (optional as defined in the 'params'). Seurat object and metadata for the library along with UMAP embeddings are saved to be used for downstream analyses.

101 **(4) Final QC summary report**  
102 Lastly, we provide a final QC summary report containing graphs and summary tables across each QC step.

103  
**Folder structure**  
104 The structure of this folder is as follows:  
105 ...
106 ...
107 ...
108 ...
109 ...
110 ...
111 ...
112 ...
113 ...
114 ...
115 ...
116 ...
117 ...
118 ...
119 ...
120 ...
121 ...
122 ...
123 ...
124 ...
125 ...
126 ...
127 ...
128 ...
129 ...
130 ...
131 ...
132 ...
133 ...
134 ...
135 ...
136 ...
137 ...
138 ...
139 ...
140 ...
141 ...
142 ...
143 ...
144 ...
145 ...
146 ...
147 ...
148 ...
149 ...
150 ...
151 ...
152 ...
153

# ‘upstream-analysis` module

---

#	steps	aim
1	Signac QC metrics	To infer QC metrics and remove low quality cells

```
### 🧪 Key QC Metrics and Suggested Thresholds (Per Cell/Nucleus)

| **Metric** | **Interpretation** | **Low-Quality Threshold** | **High-Quality Range** |
|-----|-----|-----|
| **Fragments in peaks** | Proportion of reads in peaks – reflects signal-to-noise | < 20-30% | > 30-50% |
| **TSS enrichment score** | Enrichment of reads around transcription start sites | < 4-6 | > 6-10 |
| **Nucleosome signal** | Ratio of mono- to di-/tri-nucleosome fragments (chromatin state) | > 4 | < 2-3 |
| **Total fragments** | Total number of fragments per cell. Avoid very low or very high. | < 1k or > 50k | 3k-30k |
| **Blacklist ratio** | Fraction of reads in ENCODE blacklist regions (noise -High = poor quality) | > 0.01 | < 0.01 |
| **Mono-/multi-nucleosome** | Fragment length distribution should show clear banding | Poor/no banding | Visible nucleosome bands |
| **Doublet score** | Estimate of multiplets (2+ cells) per barcode (Use to exclude multiplets) | High | Low (filtered out) |
```



# ‘upstream-analysis’ module

#	steps	aim
1	Signac QC metrics	To infer QC metrics and remove low quality cells
2	scDblFinder method	To identify doublets/multiplets

```
112 ### (2) Estimating and filtering out doublets
113
114 Popular approach of single-cell uses oil droplets or wells to isolate single cells along with barcoded beads. Depending on the cell density loaded, a proportion of reaction volumes (i.e. droplets or wells) will capture more than one cell, forming ‘doublets’ (or ‘multiplets’), i.e. two or more cells captured by a single reaction volume and thus sequenced as a single-cell artifact.
115
116 The proportion of doublets is proportional to the number of cells captured. It is common in single-cell experiments to have 10-20% doublets, making accurate doublet detection critical.
117
118 Doublets are prevalent in single-cell sequencing data and can lead to artifactual findings. We will use a computational approach to calculate and remove doublets from the library. Here, we use [ScDblFinder](https://bioconductor.org/packages/devel/bioc/vignettes/scDblFinder/inst/doc/scDblFinder.html) method for identifying doublets/multiplets in single-cell data.
119
120 The `seurat_obj_raw.rds` object from step (1) is used for this step.
121 - Summary table with doublet metrics and doublets prediction plot are generated.
```



# `upstream-analysis` module

#	steps	aim
1	Signac QC metrics	To infer QC metrics and remove low quality cells
2	scDblFinder method	To identify doublets/multiplets (step 2 is optional during final filtering)
3	Merge and filter data	To create clean object for downstream analysis
4	Final QC summary report	To summarize all results in one

## ### (3) Merging filtered data

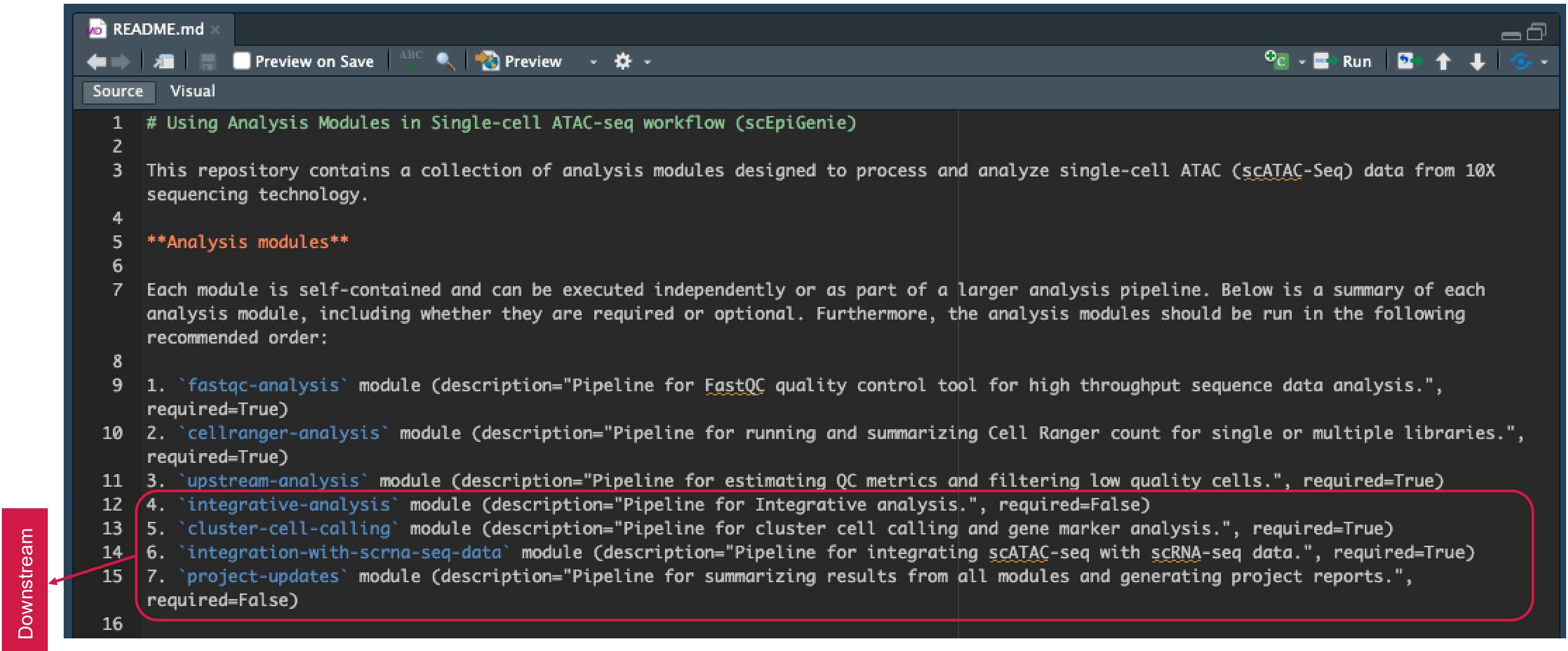
Next, we merge count matrices from steps (1-2) after filtering out low quality cells and doublets (optional as defined in the `params`). Seurat object and metadata for the library along with UMAP embeddings are saved to be used for downstream analyses.

## ### (4) Final QC summary report

Lastly, we provide a final QC summary report containing graphs and summary tables across each QC step.



# scEpiGenie analysis modules

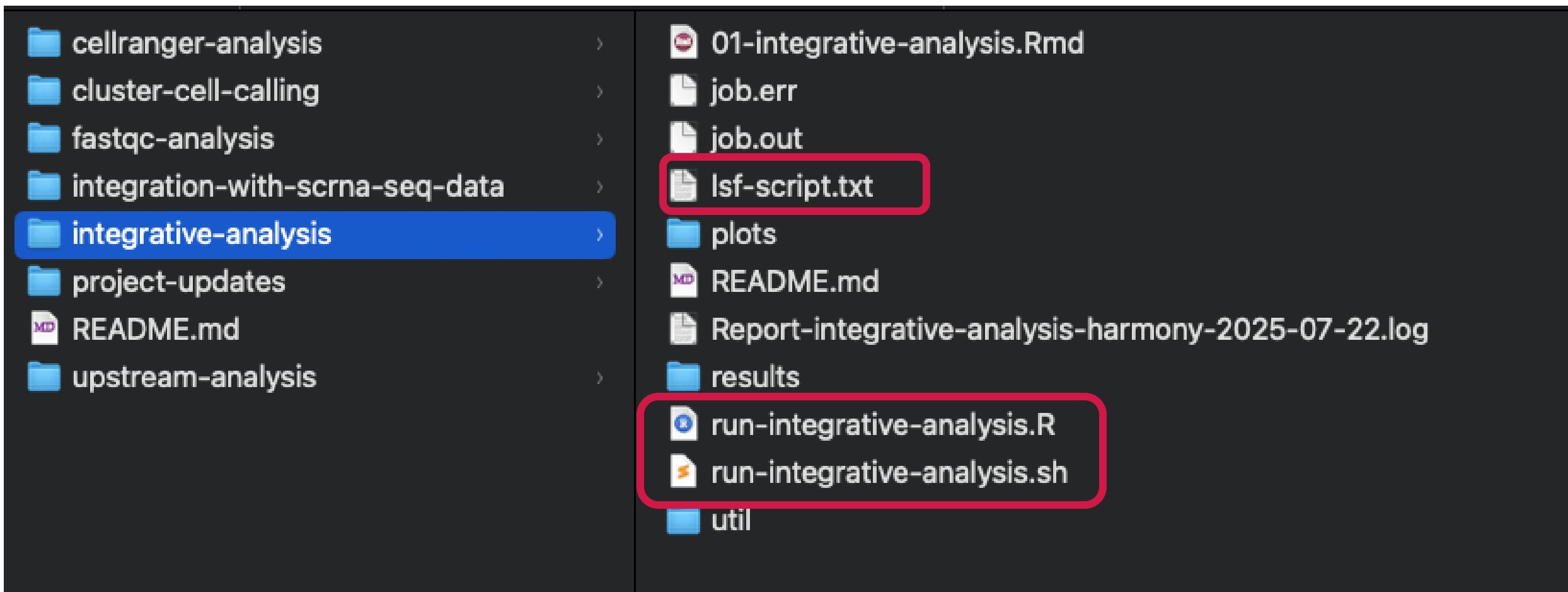


```
1 # Using Analysis Modules in Single-cell ATAC-seq workflow (scEpiGenie)
2
3 This repository contains a collection of analysis modules designed to process and analyze single-cell ATAC (scATAC-Seq) data from 10X sequencing technology.
4
5 **Analysis modules**
6
7 Each module is self-contained and can be executed independently or as part of a larger analysis pipeline. Below is a summary of each analysis module, including whether they are required or optional. Furthermore, the analysis modules should be run in the following recommended order:
8
9 1. `fastqc-analysis` module (description="Pipeline for FastQC quality control tool for high throughput sequence data analysis.", required=True)
10 2. `cellranger-analysis` module (description="Pipeline for running and summarizing Cell Ranger count for single or multiple libraries.", required=True)
11 3. `upstream-analysis` module (description="Pipeline for estimating QC metrics and filtering low quality cells.", required=True)
12 4. `integrative-analysis` module (description="Pipeline for Integrative analysis.", required=False)
13 5. `cluster-cell-calling` module (description="Pipeline for cluster cell calling and gene marker analysis.", required=True)
14 6. `integration-with-scrna-seq-data` module (description="Pipeline for integrating scATAC-seq with scRNA-seq data.", required=True)
15 7. `project-updates` module (description="Pipeline for summarizing results from all modules and generating project reports.", required=False)
16
```

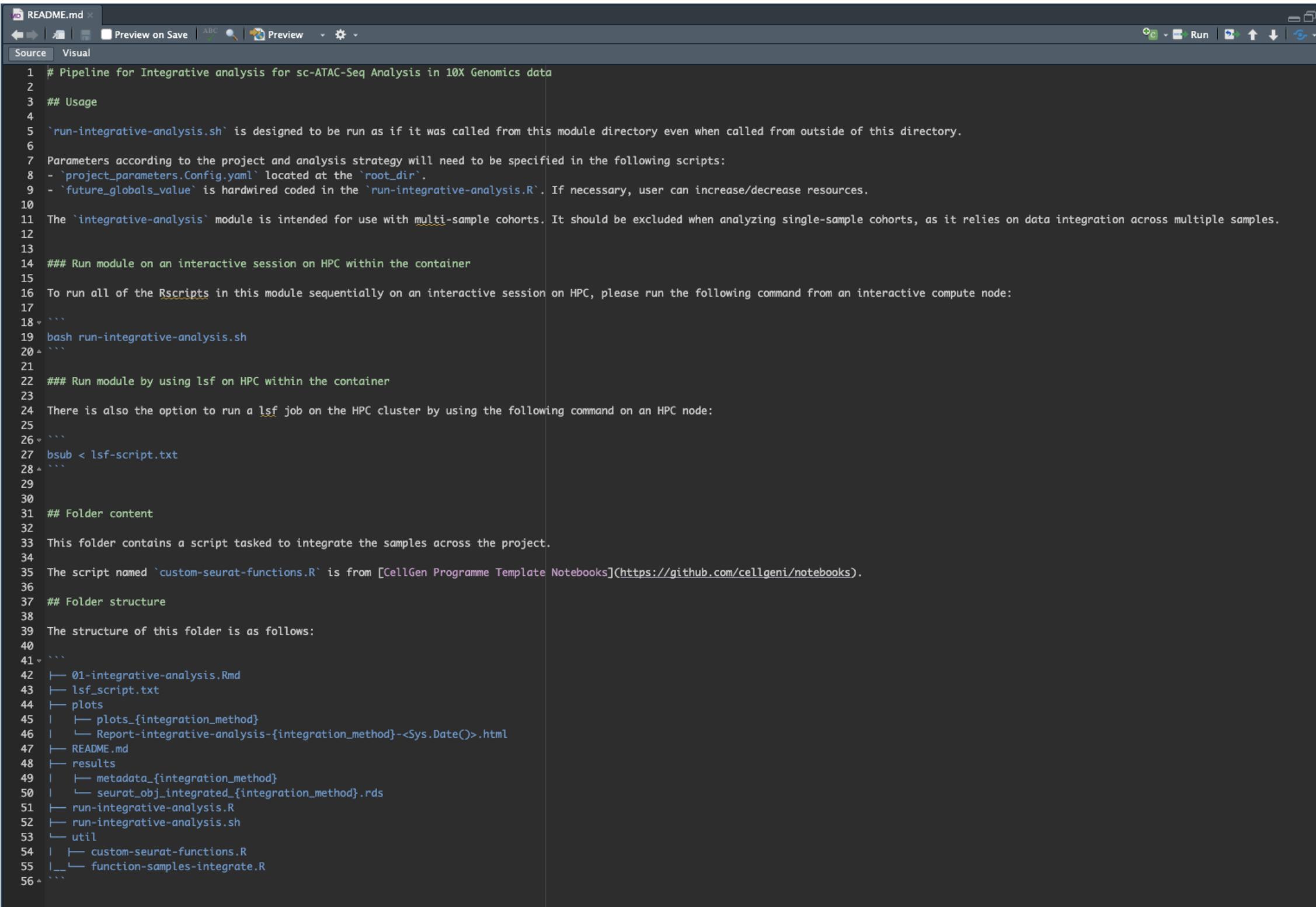


# `integrative-analysis` module

description="Pipeline for Integrative analysis.", required=False.



# `integrative-analysis` module



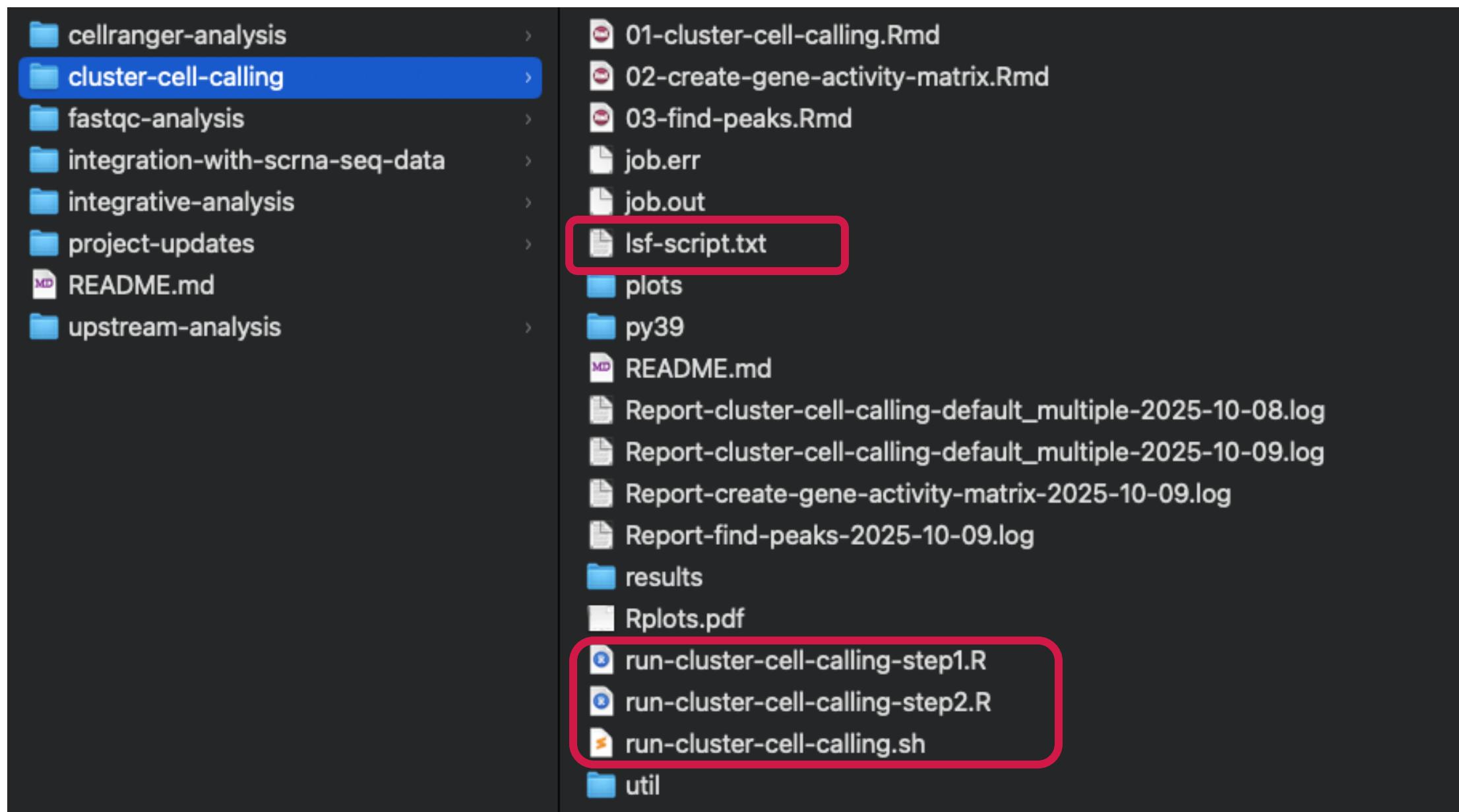
The screenshot shows a code editor window with the file "README.md" open. The content of the file is a detailed description of the `run-integrative-analysis.sh` script. It includes instructions for usage, parameters, and running the script on an HPC cluster. It also describes the folder structure and contents of the module.

```
1 # Pipeline for Integrative analysis for sc-ATAC-Seq Analysis in 10X Genomics data
2
3 ## Usage
4
5 `run-integrative-analysis.sh` is designed to be run as if it was called from this module directory even when called from outside of this directory.
6
7 Parameters according to the project and analysis strategy will need to be specified in the following scripts:
8 - `project_parameters.Config.yaml` located at the `root_dir`.
9 - `future_globals_value` is hardwired coded in the `run-integrative-analysis.R`. If necessary, user can increase/decrease resources.
10
11 The `integrative-analysis` module is intended for use with multi-sample cohorts. It should be excluded when analyzing single-sample cohorts, as it relies on data integration across multiple samples.
12
13
14 ### Run module on an interactive session on HPC within the container
15
16 To run all of the Rscripts in this module sequentially on an interactive session on HPC, please run the following command from an interactive compute node:
17
18 ```
19 bash run-integrative-analysis.sh
20 ```
21
22 ### Run module by using lsf on HPC within the container
23
24 There is also the option to run a lsf job on the HPC cluster by using the following command on an HPC node:
25
26 ```
27 bsub < lsf-script.txt
28 ```
29
30
31 ## Folder content
32
33 This folder contains a script tasked to integrate the samples across the project.
34
35 The script named `custom-seurat-functions.R` is from [CellGen Programme Template Notebooks](https://github.com/cellgeni/notebooks).
36
37 ## Folder structure
38
39 The structure of this folder is as follows:
40
41 ```
42 ├── 01-integrative-analysis.Rmd
43 ├── lsf_script.txt
44 ├── plots
45 │   ├── plots_{integration_method}
46 │   └── Report-integrative-analysis-{integration_method}-<Sys.Date()>.html
47 └── README.md
48 └── results
49     ├── metadata_{integration_method}
50     └── seurat_obj_integrated_{integration_method}.rds
51 └── run-integrative-analysis.R
52 └── run-integrative-analysis.sh
53 └── util
54     └── custom-seurat-functions.R
55 └── function-samples-integrate.R
56 ````
```



# `cluster-cell-calling` module

description="Pipeline for cluster cell calling and differentially accessible peaks analysis.", required=True.



# `cluster-cell-calling` module

```

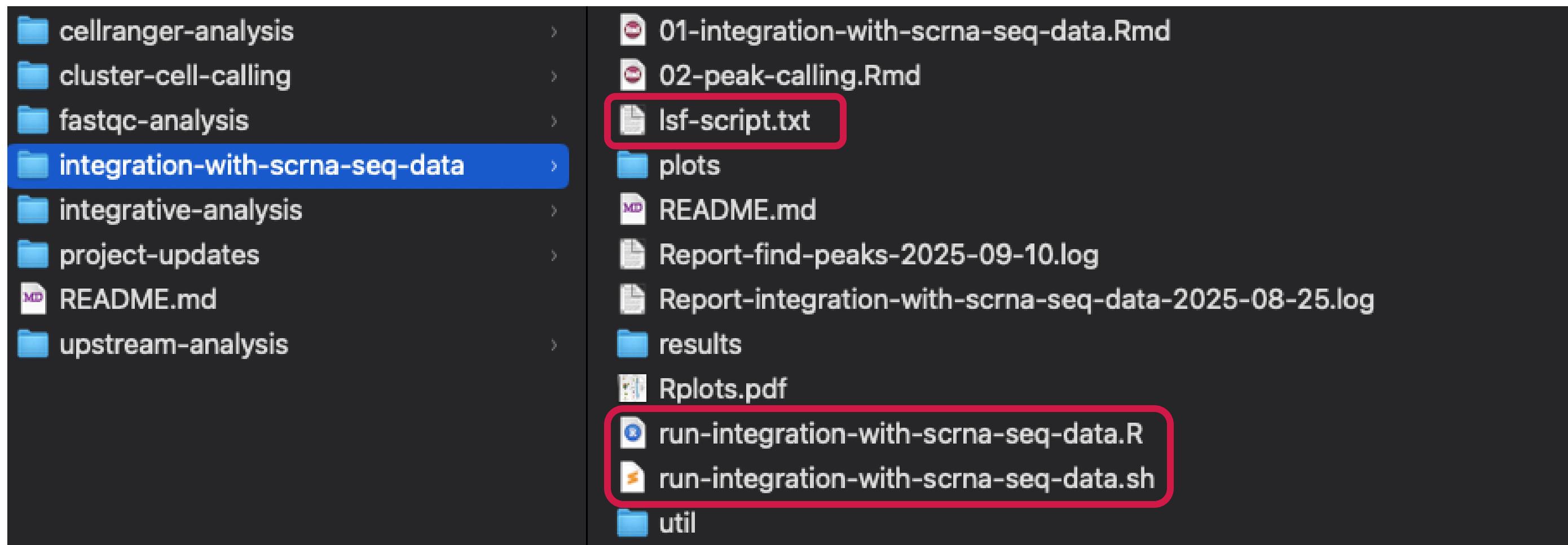
1 # Pipeline for cluster cell calling and differentially accessible peaks analysis for sc-ATAC-Seq Analysis in 10X Genomics data
2
3 ## Usage|
4
5 `run-cluster-cell-calling.sh` is designed to be run as if it was called from this module directory even when called from outside of this directory.
6
7 Parameters according to the project and analysis strategy will need to be specified in the following scripts:
8 - `project_parameters.Config.yaml` located at the `root_dir`.
9 - `run-cluster-cell-calling.sh`: comment in/out according to which step user wants to run, i.e., `run-cluster-cell-calling-step1.R` and/or `run-cluster-cell-calling-step2.R`.
10 - `future_globals_value` is hardwired coded in the `run-cluster-cell-calling-step1.R` and `run-cluster-cell-calling-step2.R`. If necessary, user can increase/decrease resources.
11
12
13 Please note that for cohorts with multiple samples, the object generated by the `integrative-analysis` module will be used by default. For single-sample cohorts, however, the merged object from the `upstream-analysis` module should be used instead. You should specify the module object to use in the `project_parameters.Config.yaml` file accordingly.
14
15
16 ### Run module on an interactive session on HPC within the container
17
18 To run all of the Rscripts in this module sequentially on an interactive session on HPC, please run the following command from an interactive compute node:
19 ...
20+
21 bash run-cluster-cell-calling.sh
22+
23
24 ### Run module by using lsf on HPC within the container
25
26 There is also the option to run a lsf job on the HPC cluster by using the following command on an HPC node:
27 ...
28+
29 bsub < lsf-script.txt
30+
31
32 ## Folder content
33 This folder contains a script tasked to calculate clusters and find differentially accessible peaks for each cluster across the project.
34
35 ## Analysis strategy:
36 We recommend the user to follow the following steps for running the current module:
37
38 - Step (1) `run-cluster-cell-calling-step1.R`: At first, the `01-cluster-cell-calling.Rmd` should be run for a set of resolutions by default (option for `default_multiple`). We recommend to run first by default and then explore a customized list of resolutions (if these are not provided in the list already; option for `custom_multiple`).
39 - Step (2) `run-cluster-cell-calling-step2.R`: After inspection of the first round of results, the single resolution that fits best the data can be provided and used to run the `03-find-peaks.Rmd`.
40
41 For more information, see [Signac clustering](https://stuartlab.org/signac/articles/mouse\_brain\_vignette).
42
43 In order to interpret the clustering and decide properly on the resolution, we should consider tissue type, total number of cells in the dataset, and expected known cell types in there (i.e., tissue ecosystem). That would help to determine the correct number of clusters in a biologically meaningful way, considering known cell types and avoiding splitting clusters into unstable, small ones. That can also help to explore smaller clusters that might contain unknown or disease/patient-specific clusters that are still worth considering and investigating from the research/clinical perspective.
44
45 After inspection of the two rounds of results in the `cluster-cell-calling` module, user can remove clusters if necessary. This is relevant to projects in which we need to remove clusters from the object. This is the case, e.g., in PDX projects, there might be both human and mouse clusters identified after the `03-find-peaks.Rmd` step of the the `cluster-cell-calling` module. In this case, we recommend the user to run the `contamination-remove-cells-analysis` module that allows to remove clusters, repeat normalization and integration steps. This object can then be used for cell type annotation or other type of analysis.
46
47
48 ## Folder structure
49
50 The structure of this folder is as follows:
51 ...
52+
53 |-- 01-cluster-cell-calling.Rmd
54 |-- 02-create-gene-activity-matrix.Rmd
55 |-- 03-find-peaks.Rmd
56 |-- lsf-script.txt
57 |-- plots
58 |   |-- 01_cluster_cell_calling_{resolution}
59 |   |-- 03_find_markers
60 |   |-- Report_cluster_cell_calling_{resolution}_{<Sys.Date()>.html}
61 |   |-- Report_cluster_cell_calling_{resolution}_{<Sys.Date()>.pdf}
62 |   |-- Report_create-gene-activity-matrix_{resolution}_{<Sys.Date()>.html}
63 |   |-- Report_create-gene-activity-matrix_{resolution}_{<Sys.Date()>.pdf}
64 |   |-- Report_find_peaks_{Sys.Date()>.html}
65 |   |-- Report_find_peaks_{Sys.Date()>.pdf}
66 |-- README.md
67 |-- results
68 |   |-- 01_cluster_cell_calling_{resolution}
69 |   |-- 02_create_gene_activity_matrix
70 |   |-- 03_find_peaks
71 |-- run-cluster-cell-calling-step1.R
72 |-- run-cluster-cell-calling-step2.R
73 |-- run-cluster-cell-calling.sh
74 |-- util
75 |   |-- calculate-fold-change.R
76 |   |-- export-cluster-peaks.R
77 |   |-- function-cluster-cell-calling.R
78 |   |-- run-clusterwise_G0_enrichment.R

```



# `integration-with-scrna-seq-data` module

description="Pipeline for integrating scATAC-seq with scRNA-seq data.", required=True.



# `integration-with-scrna-seq-data` module

```

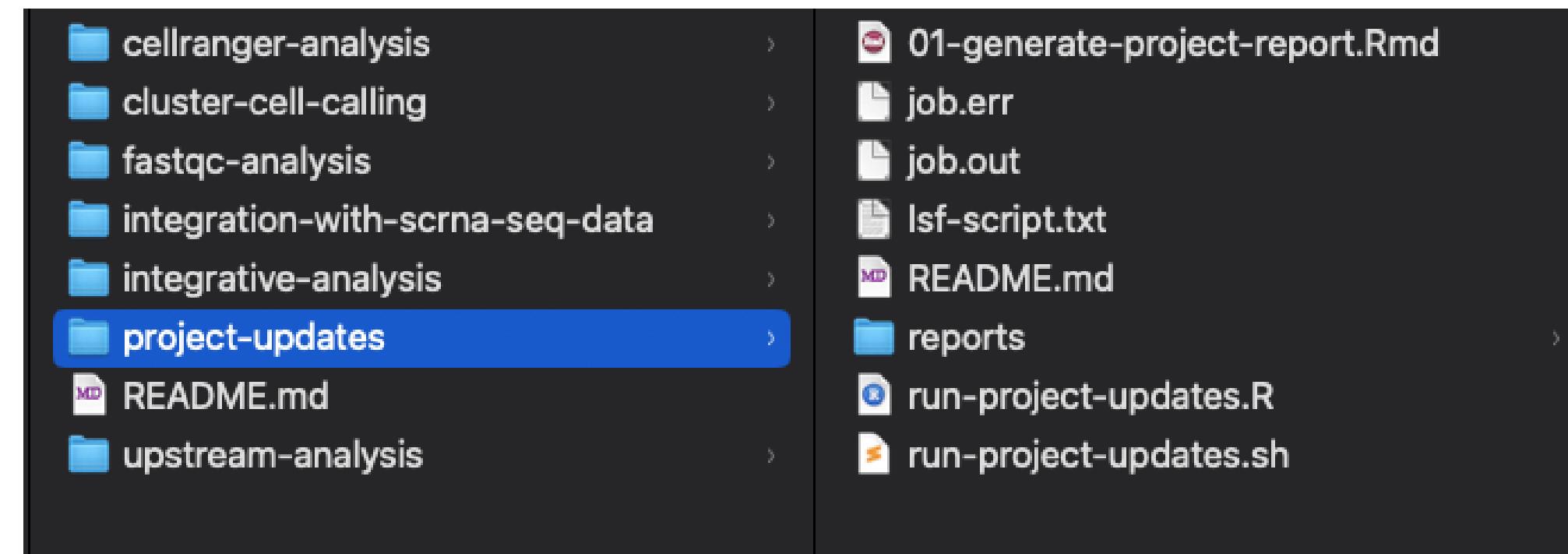
README.md x
Source Visual
1 # Pipeline for Peak calling per cell type following integration with scRNA-seq data for scATAC-Seq Analysis in 10X Genomics data
2
3
4 ## Usage
5
6 `run-integration-with-scrna-seq-data.sh` is designed to be run as if it was called from this module directory even when called from outside of this directory.
7
8 Parameters according to the project and analysis strategy will need to be specified in the following scripts:
9 - `project_parameters.Config.yaml` located at the `root_dir`
10 - `future_globals_value` is hard-coded in each `run-integration-with-scrna-seq-data.R` script for each method. If necessary, user can increase/decrease resources.
11
12 We provide a fixed color palette for annotating cell types. Please ensure that the cell type names in your customized cell type marker reference list and/or reference dataset (if you use any) exactly match those in the `figures/palettes/cell_types_palette.tsv`. If any cell types are missing, please submit an [issue](https://github.com/stjude-dnb-bincore/sc-rna-seq-snap/issues) with the list of missing types, and we will add them to the palette. Alternatively, users are welcome to use their own `cell_types_palette.tsv`, as long as the column names ('cell_type_names' and 'hex_codes'), file name, and format match those in `./figures/palettes/cell_types_palette.tsv`. We recommend that users use the same color palette files across pipelines and projects, i.e., same files as for the `sc-rna-seq-snap` pipeline.
13
14
15 ### Run module on an interactive session on HPC within the container
16
17 To run all of the Rscripts in this module sequentially on an interactive session on HPC, please run the following command from an interactive compute node:
18
19 ...
20 bash run-integration-with-scrna-seq-data.sh
21 ...
22
23 ### Run module by using lsf on HPC within the container
24
25 There is also the option to run a lsf job on the HPC cluster by using the following command on an HPC node:
26
27 ...
28 bsub < lsf-script.txt
29 ...
30
31 ## Folder content
32
33 This folder contains a script tasked to perform a data transfer method in the context of scATAC-seq to:
34
35 - Classify cells measured with scATAC-seq based on clustering results from scRNA-seq
36 - Co-embed scATAC-seq and scRNA-seq data
37
38 There is also a script to perform peak calling and gene ontology analysis per cell type.
39
40
41 ## Folder structure
42
43 The structure of this folder is as follows:
44
45 ...
46 ├── 01-integration-with-scrna-seq-data.Rmd
47 ├── 02-peak-calling.Rmd
48 └── lsf_script.txt
49 └── plots
50   └── 01_integration_with_scrna_seq_data
51     └── 02_peak_calling
52       └── Report-integration-with-scrna-seq-data-<Sys.Date()>.html
53       └── Report-integration-with-scrna-seq-data-<Sys.Date()>.pdf
54       └── Report-peak-calling-<Sys.Date()>.html
55       └── Report-peak-calling-<Sys.Date()>.pdf
56 └── README.md
57 └── results
58   └── 01_integration_with_scrna_seq_data
59     └── 02_peak_calling
60 └── run-integration-with-scrna-seq-data.R
61 └── run-integration-with-scrna-seq-data.sh
62 └── util
63   └── co-embedding-cells.R
64   └── function_process_fragments.R
65   └── function-cell-type-fractions.R
66 ...
67

```



# `project-updates` module

description="Pipeline for summarizing results from all modules and generating project reports.", required=False.



# Future analysis modules

---

- Motif analysis with Signac
- Transcription factor footprinting
- Building trajectories with Monocle 3
- Finding co-accessible networks with Cicero



## Next steps

---

-  **Pipeline development (and testing phase)**
-  **Testing phase and code review process**
-  **Launching: Beta Release 1.0.0**
-  **Validation phase by using other datasets/experiments est. by January 2026**





# Ask me questions!



Tonia  
**Antonia Chroni, PhD**  
she/her/hers

Senior Bioinformatics Research Scientist  
New York, New York



DEMO

